Hochschule Mittweida (FH)
University of Applied Sciences

Fakultät IT/ET

Diplomarbeit

im Studiengang Multimediatechnik

Analyse und Konzeptionierung einer Streaminglösung für die Präsentation von Audio- und Videoinhalten in einem intermedialen Redaktionssystem, als Grundlage für eine Online-Plattform der Hochschule Mittweida

eingereicht von: Heiko Milker <heiko.milker@multimediatechniker.de>

eingereicht am: 29. Januar 2010

Betreuung: Herr Prof. Dr. Michael Hösel, Hochschule Mittweida
Herr Dipl.-Ing. Peter Lubosch, Hochschule Mittweida

Danksagung

Zu Beginn möchte ich meinen besonderen Dank an Personen richten, die mir während der Erstellung dieser Arbeit mit Rat und Tat zur Seite standen. Ohne diese Personen wäre ein letztendlicher Abschluss dieser Arbeit nicht möglich gewesen.

Ich danke meiner Mutter für die Zeit und die Mühen die sie aufgewendet hat um mich auf die Tiefen der deutschen Sprache aufmerksam zu machen.

Ich danke besonders der Patricia, die ebenfalls den Kampf mit dem Fehlerteufel gewagt und mir so manchen Fehler fast um die Ohren gehauen hat.

Ein Dank richtet sich auch an den Herr Eckelt, der mir mit seinem Fachwissen und seinem technischen Know-how viele gedruckte Exemplare zum redigieren dieser Arbeit zur Verfügung stellte und schließlich auch den finalen Druck erledigte.

Ebenfalls möchte ich meinen Betreuern den Herren Prof. Dr. Michael Hösel und Dipl.-Ing. Peter Lubosch danken, die mir mit viel Geduld und so manchem Hinweis stets zur Seite standen.

Bibliografische Beschreibung

Milker, Heiko:
Analyse und Konzeptionierung einer Streaminglösung für die Präsentation von Audio- und Videoinhalten in einem intermedialen Redaktionssystem, als Grundlage für eine Online-Plattform der Hochschule Mittweida - 2009. - 103 Seiten
Mittweida, Hochschule Mittweida, Fachbereich IT/ET, Diplomarbeit, 2010

Referat:

Ziel der Diplomarbeit ist es, mögliche Lösungsansätze für die Präsentation von Audio- und Videodaten auf einer Portalseite aufzuzeigen. Nach der Vorstellung wichtiger Grundlagen und Begriffen, die das Thema Streaming betreffen, werden verschiedene Lösungsansätze und Techniken vorgestellt. Aus diesen Ansätzen werden anhand von zuvor definierten Anforderungen und der Auswertung einer Online-Umfrage mögliche Lösungswege definiert. Aus diesen Lösungsansätzen und Techniken werden daraufhin Konzepte erstellt und prototypisch umgesetzt.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung und Ziel dieser Arbeit

Das Internet und die damit verknüpften Technologien entwickeln sich rasant. Längst hat das Internet viele heimische Wohnzimmer erreicht und bietet neben nützlichen Informationen auch vermehrt multimediale Angebote wie Audio und Video. Mit diesem Trend hat das Internet schon jetzt den dritten Platz der tagesaktuellen Medien belegt und liegt damit direkt hinter dem Fernseh- und Hörfunkkonsum der Deutschen.

Mittlerweile ist es für einen Teil der deutschen Internet-Benutzer schon fast zu einer Selbstverständlichkeit geworden den Computer schon am frühen Morgen anzuschalten, E-Mails abzurufen, News-Seiten anzusurfen und die neuesten Nachrichten gebündelt auf den Monitor zu durchstöbern. Ebenfalls zeichnet sich der Trend ab, dass ein Teil der deutschen Internet-Benutzer allmorgendlich den Audio- oder Video Player startet um online Radioinhalte, Nachrichten, Videos oder die Lieblingsmusik zu konsumieren.

Durch den steigenden Gebrauch des Internets als Kommunikations- und Informationsplattform rückt auch der Wunsch nach Unterhaltung über das Medium Internet immer weiter in den Vordergrund. So waren beispielsweise im Jahre 2009 bereits rund 67 % der deutschen Erwachsenen durchschnittlich 58 Minuten am Tag online.

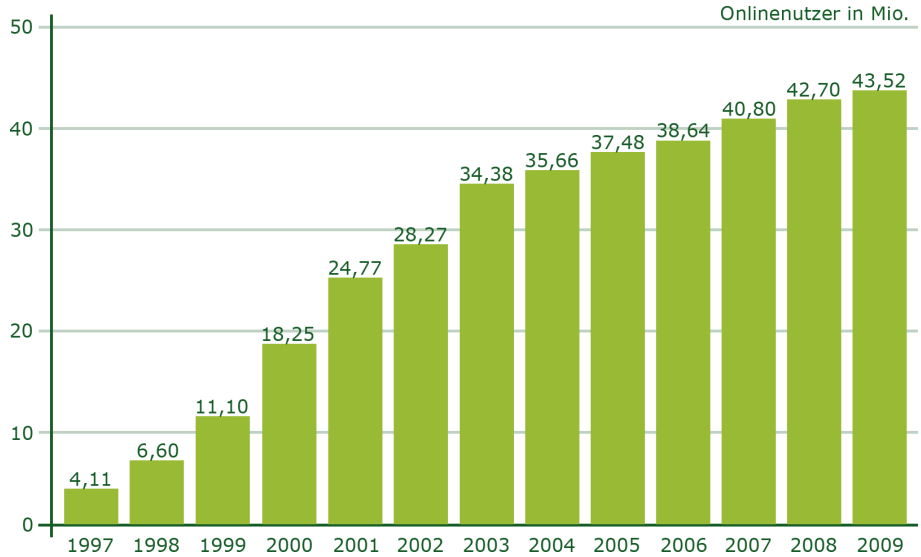


Abbildung 1: Entwicklung der Onlinenutzung in Deutschland

(ARD/ZDF-Online-Studie 1997-2009)

Natürlich steht bei der Nutzung des Internets immer noch der Abruf von Informationen an erster Stelle, jedoch steigt die Nutzung als Unterhaltungsplattform von Jahr zu Jahr an.

Im Frühjahr 2008 waren bereits 19% der Internetnutzer auf der Suche nach Unterhaltung im WWW, 2007 waren es hingegen erst 14%. Im Jahre 2009 riefen 62% (2008: 55%) aller Onliner Videos, zum Beispiel über Videoportale oder Mediatheken, ab und schauen live oder zeitversetzt Fernsehsendungen im Internet. 51% (2008: 43%) hören Audiofiles wie Musikdateien, Podcasts und Radiosendungen im Netz. (BR-ONLINE)

Diese Entwicklung in Deutschland zeigt auf, dass die klassischen Medien langfristig Gefahr laufen könnten als Informations- und Unterhaltungsmedium vom Internet verdrängt zu werden. Dabei ist die Erweiterung des Angebotes der klassischen Medien auf das Internet technisch machbar und bietet zudem viele Vorteile. Neben der Erweiterung des Services für die derzeitigen Nutzer der klassischen Medien, werden so auch einige neue Nutzer durch die Online-Angebote gewonnen.

1.1 Ziel dieser Arbeit

Ziel dieser Arbeit ist es, eine Analyse der zur Verfügung stehenden Techniken durchzuführen, sowie Konzepte für die Umsetzung einer geeigneten Streaminglösung für ein intermediales Redaktionssystems zu finden.

Das intermediale Redaktionssystem, welches neben den herkömmlichen Inhalten wie Text und Bild zukünftig auch die eigene Bereiche für Video und Audio auf dem resultierendem Portal präsentieren soll, wird die bisher einzelnen und dadurch unterschiedlichen Web-Präsentationen der Bereiche Radio, TV und Zeitung der Hochschule Mittweida vereinen. Das intermediale Redaktionssystem soll einem Benutzer die in den schon existierenden Bereichen enthaltenen Informationen auf einer neuen und einheitlichen Plattform im Internet präsentieren können und dabei den Redakteuren der einzelnen Bereiche die Arbeit durch ein einheitliches System erleichtern können.

Um das eben genannte Ziel überhaupt erreichen zu können, ist es notwendig diese Thematik in zwei Teile zu unterteilen. Auf der einen Seite stehen die Probleme der eigentlichen Portalerstellung, der Benutzer- und Bereichsverwaltung und des Designs des Redaktionssystems. Auf der anderen Seite steht das Erstellen und Entwerfen einer geeigneten Streaminglösung für das resultierende Portal. Diese Arbeit soll lediglich der Planung und Entwicklung der Streaminglösungen für ein intermediale Redaktionssystem dienen, deswegen wird in dieser Arbeit nicht auf die Erstellung und Verwaltung eines Webportales eingegangen.

1.2 Kapitelübersicht

Kapitel 2 gibt eine Einführung in die Grundbegriffe und Grundlagen der Thematik Streaming. Hier werden wichtige Techniken, Verfahren und Produkte vorgestellt und genauer erklärt. Diese Grundlagen sind Vorrausetzung um die in den nachfolgenden Kapiteln

vorgestellten Konzepte und Workarounds verstehen zu können.

Kapitel 3 enthält die Beschreibung und Auswertung einer Umfrage, die speziell für diese Arbeit durchgeführt wurde. Anhand der Umfrage wurden wichtige Informationen und Hinweise gesammelt, die in den nachfolgenden Kapiteln mit einfließen.

In **Kapitel 4** werden einige notwendige Vorüberlegungen für geeignete Lösungsansätze aufgeführt und diskutiert. Aus diesen Vorüberlegungen entstehen diverse Anforderungen, die im nächsten Kapitel beschrieben werden.

Kapitel 5 enthält verschiedene Anforderungen für eine Audio-Streaming und eine Video-Streaminglösung, die auf dem resultierenden Portal zum Einsatz kommen sollen.

Im **Kapitel 6** werden anhand der zuvor aufgestellten Anforderungen diverse Lösungsansätze für das Live- und On-Demand-Streaming vorgestellt und genauer erklärt.

Das **Kapitel 7** enthält einige Prototypische Umsetzungen der zuvor aufgeführten Lösungsansätze. In diesem Kapitel wird die Installation und Konfiguration einer kompletten Audio-Streaminglösung mittels Streaming-Server und Encoder vorgestellt, die im Anschluss in der Lage ist, Audio-Inhalte über einen MP3-Stream und einen Flash-Stream zu verbreiten.

Eine Zusammenfassung als auch die sukzessive Rekapitulation der einzelnen Abschnitte der Bearbeitung der Diplomarbeit, befinden sich in **Kapitel 8**.

2 Grundlagen und Grundbegriffe

Um überhaupt einen Lösungsansatz für eine geeignete Streaminglösung finden zu können, ist es notwendig sich im Vorfeld mit den Grundlagen rund um das Thema Streaming zu beschäftigen. Die Thematik „Streaming im Internet“ bringt viele neue Begriffe, Techniken und Verfahren mit sich, auf die in den nachfolgenden Kapiteln etwas eingegangen werden soll.

2.1 Der Begriff „Streaming“

Audio- und Video Daten werden in der Regel sequentiell, also beginnend vom Dateianfang bis zum Dateiende, abgespielt. Da es aber ein großer zeitlicher Aufwand wäre zum Beispiel einen Film erst komplett herunterzuladen und anschließend abzuspielen, ist es sinnvoll mit dem Abspielen schon während der Übertragung zu beginnen und nicht erst auf das Dateiende zu warten, wie es bei einem normalen Dateidownload der Fall wäre. (MEIßNER K.)

Von Streaming spricht man also dann, wenn Audio- und Video Daten unter Echtzeit-Anforderungen, also mit einer garantierten und geringen Verzögerung, übertragen werden und der Inhalt schon während der Übertragung genutzt werden kann.

Im Bereich der Online-Medien wird das Streamen von Audio und Video im Allgemeinen als Internet-Äquivalent zu den bekannteren Broadcasting-Technologien wie Rundfunk oder Fernsehen gesehen, also der Übertragung von Daten von einem zentralen Punkt aus zu allen Teilnehmern innerhalb eines Netzwerkes. Am Beispiel eines klassischen Radiosenders, verbreitet ein Funkmast (als zentraler Punkt gesehen) das Radiosignal an viele Teilnehmer über das Medium Luft. Bei dem Äquivalent eines Internet-Radios, ist es ein Streaming-Server der die digitalen Inhalte über das Medium Internet an die Teilnehmer verbreitet.

Um die Last eines Streaming-Servers bei einer großen Anzahl von Klienten zu verringern, gibt es zwei wichtige Methoden die zu streamenden Daten zu verteilen. Zum einen die Verteilung des Streams über einen Replication-Server und zum Anderen die Verteilung über einen Relay-Server.

2.1.1 Funktionsweise eines Streams

Da Audio- und Video Inhalte normalerweise von einem Benutzer sequentiell genutzt, also vom Anfang bis zum Ende wiedergegeben und betrachtet werden, ist es sinnvoll diese Daten auch so im Internet zu übertragen. Aus diesem Grund entwickelte und verwendet man bei der Übertragung von Audio- und Video Daten eine eigene Strategie.

Im Gegensatz zu einem klassischen Datei-Download, bei dem die Daten von Anfang bis

zum Ende heruntergeladen werden müssen und dadurch erst anschließend genutzt werden können, werden beim Streaming die Daten schubweise im sogenannten Teilstreckenverfahren (Store-and-Forward) an den Empfänger übertragen. Die so übertragenen Daten-Pakete werden im Anschluss von dem Empfänger wieder zusammengefügt. Eine Serie an zusammengehörigen Paketen wird als Stream bezeichnet.

Um zu verhindern, dass die Wiedergabe des Streams ins stocken gerät, weil zum Beispiel die verfügbare Bandbreite kurzzeitig sinkt, werden die Pakete vom Client vorgeladen. Auf der Empfängerseite werden die Pakete ab der ersten Sekunde in einem Puffer zwischengespeichert und können von dort sofort bei Bedarf entnommen und genutzt werden. Diesen Vorgang nennt man Pufferung (engl.: Buffering) und wird durch den Player angewandt, der den Stream wiedergeben soll.

2.1.2 Anforderungen an das Übertragungsmedium

Heutzutage vereint das Medium Internet die Möglichkeit Video- und Audiodateien verteilen zu können. Über das Internet kann prinzipiell jede Art von digitalen Daten übertragen werden. Die so übertragenen Informationen unterliegen lediglich den Einschränkungen des Übertragungsmediums, also der Bandbreite und Charakteristik des Internets. Kennzeichnend für die Übertragung von Audio- und Video Datenströmen sind enge zeitliche Toleranzen während der Übertragung, sowie große Datenmengen besonders bei Video-Streams und die daraus resultierend hohe benötigte Bandbreite.

Um den kontinuierlichen Transport von Informationen in nahezu Echtzeit garantieren zu können muss der Übertragungskanal unterschiedlichen Anforderungen etsprechen. Dieser sollte eine möglichst hohe und konstante Bandbreite mit geringer Verzögerungszeit aufweisen und die Häufigkeit und Charakteristik von eventuellen Übertragungsfehlern sollte bekannt sein.

2.1.3 Anforderungen an einen Stream

Das Übertragungsmedium Internet bietet zwar die Möglichkeit digitale Daten zu verbreiten, jedoch unterliegen solche Übertragungen dabei gewissen Einschränkungen. Während einer solchen Übertragung haben Faktoren wie die Bandbreite der zur Verfügung stehenden Internetverbindung oder die verwendeten Übertragungsprotokolle einen großen Einfluß auf den Erfolg der Übertragung. Die Menge an Daten, die im Normalfall bei einer Audio- und Videoübertragung anfallen, sind meist sehr groß und liegen gerade im Audiobereich als analoges Signal vor.

Um solche Daten über ein digitales Medium wie das Internet überhaupt übertragen zu können, müssen diese Daten im Vorfeld digital umgewandelt, in ein passendes Format gebracht und auf eine Dateigröße reduziert werden, welche sich streamen lässt. Bei der

Thematik Streaming spricht man hier von Encoding und Transkodierung, also die Kompression und Kodierung der Ursprungsdaten in ein für das Streaming geeignetes Format.

2.1.4 Die Wandlung zu einem Stream

Um ein Musikstück aus einer analogen Audioquelle oder das gesprochene Wort eines Moderators über ein Mikrofon mit möglichst geringem Qualitätsverlust über eine Internetverbindung übertragen zu können, müssen die Daten vor dem Versenden in ein für das Streaming geeignetes Format gewandelt und der begrenzten Bandbreite der Internetverbindung entsprechend komprimiert werden.

Diese Aufgabe der Wandlung eines analogen in ein digitales Signal übernehmen sogenannte Encoder. Diese Encoder können entweder hardwarebasierend oder in Verbindung mit einem Rechner softwarebasierend sein. Encoder wandelt dabei das analoge Signal in ein speziell für das Streaming geeignetes Format, man spricht hier auch von Containerformaten, um. Zusätzlich bieten die meisten Encoder noch die Möglichkeit die Parameter des so entstandenen digitalen Signals, wie zum Beispiel die Qualität, wieviele Kanäle (mono/stereo), die Bitrate oder die Samplingfrequenz zu verändern.

Encoder die einen solchen Workaround bieten, werden bei dem Thema Streaming auch Sourceclients genannt.

Nach dem ein solcher Encoder die Daten kodiert und komprimiert hat, wird der so entstandene Datenstrom an einen Streaming-Server weitergeleitet. Dieser Streaming-Server hat die Aufgabe, den Datenstrom in kleine Pakete zu teilen und an die zu ihm verbundenen Benutzer zu senden.

Für diese Art der Übertragung wurde neben den proprietären Protokollen einiger Hersteller ein eigens dafür optimiertes und standardisiertes Übertragungsprotokoll entwickelt, dass Real-Time Transport Protocol und das Real-Time Streaming Protocol (RTP/RSTP).

2.2 Grundlegende Streamingmöglichkeiten

Prinzipiell unterscheidet man bei der Thematik Streaming, neben der Art der Daten (Video und/oder Audio), zwischen zwei grundsätzlich verschiedene Verfahren, dem Live-Streaming und dem On-Demand-Streaming. Zusätzlich zu diesen zwei Verfahren existieren noch zwei streamingähnliche Verfahren, der normalen Download und der Progressiven Download.

In den folgenden Kapiteln werden diese vier Verfahren erläutert.

2.2.1 Download

Die einfachste und zugleich älteste Methode über das Internet Daten zu übertragen, ist der Dateidownload. Dieses Verfahren wird schon von Anfang an im Internet benutzt, um zum Beispiel Bilder oder andere Text-Dateien zu einem Benutzer zu übertragen. In dem heutigen Zeitalter, indem auch Musik und Videos über das Internet übertragen werden können, wird auch der normale Dateidownload dazu benutzt um solche Dateien zu einem Benutzer zu übertragen.

Da diese Methode der Datenübertragung schon altbewährt ist, wird von Seiten der Anbieter keine spezielle Technik oder Software benötigt um Audio- oder Videodateien zum Download zur Verfügung stellen zu können. Es ist lediglich ein Webserver oder ein FTP-Server von Nöten. Der Benutzer lädt die Dateien herunter und kann diese dann mit einer Software seiner Wahl so oft abspielen wie er möchte.

Da bei einem Download die Audio- oder Videodaten erst nach dem vollständigen Download bei dem Nutzer wiedergegeben werden können, spielen Parameter wie die Bitrate, die Länge oder die Übertragungsbandbreite eine eher untergeordnete Rolle. Ausschließlich die Übertragungsdauer verlängert oder verkürzt sich.

Der Einsatz eines normalen Dateidownloads als Übertragungsmöglichkeit für Audio- oder Videosequenzen kann durchaus sinnvoll sein. Vorteile dieses Verfahrens sind, dass diese Sequenzen so auch in höchster Qualität übertragen werden können, der Nutzer die freie Wahl hat mit welcher Software er diese Sequenzen abspielt und ihm die Möglichkeit geboten wird die Inhalte offline und mit anderen Geräten wiederzugeben. Sinnvoll ist es, ein bestehendes Streamingangebot um diese Möglichkeit zu erweitern und das Angebot somit zusätzlich in einer höheren Qualität anzubieten.

2.2.2 Progressiver Download

Der Progressive Download stellt eine Art Kompromiss und Zwischenschritt zwischen einem normalen Download und dem On-Demand Streaming dar. Wie bei einem Download oder dem On-Demand Streaming liegen die zu übertragenden Inhalte auf einem Server und können vom Benutzer über einen Player angefordert werden. Bei einem Progressivem Download werden die Daten genauso wie bei einem normalen Download über das HTTP-Protokoll heruntergeladen. Dabei werden die Daten jedoch in einen Puffer (Zwischenspeicher) abgelegt. Wurden ausreichend viele Daten in den Puffer eines Players geladen, kann der Benutzer diese sofort betrachten während im Hintergrund die noch fehlenden Daten nachgeladen werden.

Dieses Verfahren wird sehr häufig für die Verbreitung von Video-Clips über Webseiten eingesetzt, da man so ohne großen Aufwand und ohne spezielle Technik, wie zum Bei-

spiel einen Streaming-Server, Daten verbreiten kann. Ein Nachteil dieses Verfahrens ist jedoch, dass nur der bereits fertig heruntergeladenen Teil eines Videos betrachtet und nur innerhalb dieses Teiles navigiert (vorgespult/zurückgespult) werden kann.

2.2.3 On-Demand-Streaming

Das On-Demand-Streaming ist dem Live-Streaming nahezu äquivalent. Der einzigste Unterschied zu dem Live-Streaming ist, dass die zu versendeten Daten bereits auf einem Server hinterlegt sind und vom Benutzer speziell angefordert werden. Die Daten werden also nicht zum selben Zeitpunkt versendet indem sie gewonnen wurden, sondern werden erst dann gesendet wenn ein potentielle Empfänger sie anfordert.

Da die Streaming-Daten bereits auf einem Server hinterlegt sind, hat diese Art von Streaming einen gewissen Bedarf an Speicherplatz. Je nach der Art, ob Audio- und/oder Videodaten als auch Qualität der Daten, fällt dieser Bedarf unterschiedlich groß aus.

Ein großer Vorteil bei dem On-Demand-Streaming ist die Möglichkeit innerhalb des kompletten Angebotes Hin und Her oder Vor- und Zurückspulen zu können. Im Gegensatz zu einem Progressivem Download müssen die Daten nicht erst komplett heruntergeladen werden um innerhalb z.B. eines Videos nach vorn springen zu können. Der Streaming-Server liefert die Daten ab dem Zeitpunkt den der Benutzer anfordert, somit ist der Benutzer in der Lage diese nach kurzer Pufferung sofort benutzen zu können.

Jedoch muss bei jedem Sprung oder Spulvorgang der Wiedergabepuffer im Player erneut gefüllt werden, dies dauert je nach Verbindungsgeschwindigkeit und Qualität des Streams mehrere Momente und führt zu unschönen Pausen.

2.2.4 Live-Streaming

Das Live-Streaming bietet einem Anbieter die Möglichkeit Audio- und Videoinhalte mit nur geringer Verzögerung zu einem Benutzer zu übertragen, nachdem die Inhalte erstellt/gesendet wurden. Klassische Radiosender nutzen dieses Verfahren um ihr schon existierendes terrestriches Angebot ebenfalls über das Medium Internet verbreiten zu können.

Durch die geringen Betriebskosten und das kostengünstige Übertragungsmedium Internet haben sich viele neue Radiosender gegründet, die ausschließlich auf diese Art und Weise ihr Programm über das Medium Internet anbieten und verbreiten. Im Gegensatz zu einer On-Demand-Streaminglösung fordert der Benutzer keinen speziellen Inhalt an, sondern allein der Sender entscheidet welcher Inhalt übermittelt wird.

Bei dem Live-Streaming können die Daten in nahezu Echtzeit (Realtime) zu einem Benutzer übertragen werden, dass heißt die Inhalte können nahezu zeitgleich mit deren Er-

stellung an einen Benutzer geschickt werden. Die Inhalte eines solchen Streams liegen also nicht statisch auf einem Server, sondern kommen zum Beispiel von einer Webcam, einem Mikrofon, dem Ausgang einer Soundkarte oder der Sendeschiene eines Radiosenders.

Um einen solchen Echtzeit-Workaround erstellen zu können, benötigt diese Streaminglösung spezielle Techniken und Protokolle. Für die Kodierung und Komprimierung der Daten werden spezielle Encoder, sogenannte Live-Encoder, benötigt. Ebenfalls werden spezielle Server verwendet, die in der Lage sind die Daten in nahezu Echtzeit zu übertragen. Ein solcher Echtzeittransport von Daten ist nicht mit dem HTTP-Protokoll zu bewerkstelligen, die Protokolle die bei diesem Streamingverfahren zum Einsatz kommen heißen RTP/RTSP und RTCP.

Ein Vorteil dieses Verfahrens ist neben der Echtzeitauslieferung, dass die Daten nicht erst auf einem Datenträger gespeichert werden müssen. Ein Nachteilig ist, dass ein navigieren durch das Material (Spulen) nicht möglich ist.

2.3 Netzwerkstrategien beim Streaming

Die Digitalisierung und die Entstehung moderner Computernetzwerke bieten, im Gegensatz zu einer klassischen Rundfunkverteilung über Funkfrequenzen, individuelle und bidirektionale Kanäle an. Seitdem die Leistungsfähigkeit dieser Kanäle auch in den Privathaushalten ausreicht um digitale Video- und Audiosignale zu verarbeiten, werden sie zunehmend als Alternative zu der herkömmlichen Programmverteilung genutzt. Dabei bietet sich das Internet anhand seiner großen Teilnehmeranzahl als Massenmedium an.

Im Gegensatz zu der klassischen Verteilung von Informationen über den Rundfunk, bei dem die Informationen ungezielt an die Hörer verbreitet werden, wird bei der Audio- und Videoübertragung über das Internet genau dieser Ansatz nicht mehr verwendet. (WEGNER R. : S. 82/83)

Das Internet bedient sich drei grundsätzlichen Netzwerkstrategien um die zu übertragenen Daten zum Benutzer transferieren zu können. Diese Netzwerkstrategien heißen Broadcast (One-to-All), Multicast (One-to-Many) oder Unicast (One-to-One) Verfahren und werden in den folgenden Kapiteln erläutert.

2.3.1 Unicast Strategie

Die Unicast Strategie, oder auch Einzelverbindung genannt, entwickelte sich mit der Digitalisierung von Daten und dem Aufkommen von Computernetzwerken. Dieses Verbindungsverfahren verfügt bereits über bidirektionale Kanäle und wird auch als Punkt-zu-Punkt-Verbindung bezeichnet. Im Gegensatz zur klassischen Programmverteilung über eine Funkfrequenz, stehen bei dieser Art der Verbindung Rückantwortkanäle zur Verfügung.

Dieses Verbindungsverfahren hat jedoch einen entscheidenden Nachteil. Die Netzlast und

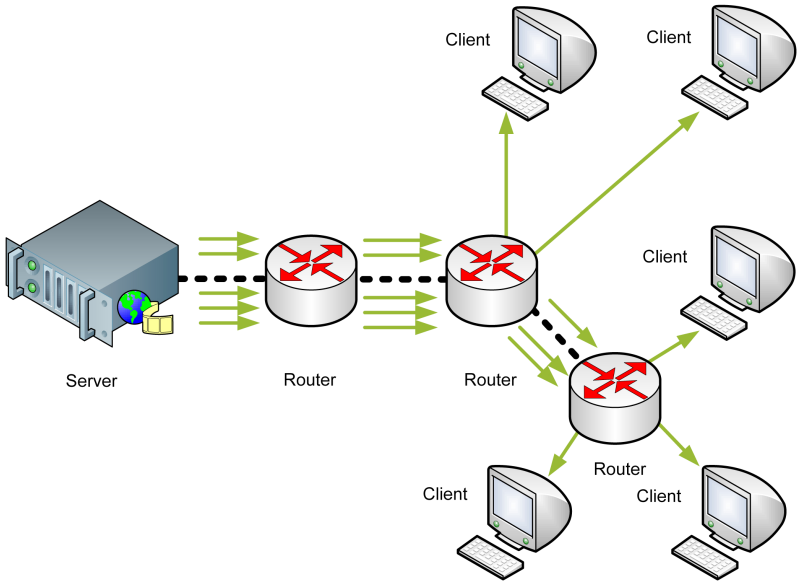


Abbildung 2: Darstellung des Unicastverfahrens

die Serverlast steigen proportional mit der Anzahl der Benutzer, da mit steigender Anzahl der Benutzer auch mehr Punkt-zu-Punkt Verbindungen aufgebaut und somit auch wesentlich mehr Daten übertragen werden müssen. Die benötigte Bandbreite multipliziert sich also mit der Zahl der Benutzer.

2.3.2 Multicast Strategie

Während bei der Unicast Strategie eine Verbindung zu nur einer IP-Adresse eines Rechners aufgebaut werden kann, wird bei dem Multicast Strategie eine Verbindung zu einer ganzen Gruppe von Rechnern aufgebaut. Diese Rechner befinden sich in einem speziell für diesem Fall reserviertem IP-Adressraum und die dazugehörigen IP-Adressen der Rechner werden als Multicast-Adressen bezeichnet. Diese Netzwerkstrategie bezeichnet man auch als Punkt-zu-Gruppe-Verbindung oder Mehrpunktverbindung. Im Gegensatz zu der Unicast Strategie, wo die Intelligenz des Verbindungsaufbaues zum größten Teil bei dem Server und dem Clienten liegt, wird bei diesem Verfahren die Intelligenz teilweise auf sogenannte Multicast-Router verschoben. Dabei bauen diese Router immer nur dann eine Verbindung zu einem anderem Multicast-Router auf, wenn ein Client Interesse an dem Streaming-Signal bekundet. In der Verbindungskette von mehreren Routern, dient hier lediglich der Router, der dem oder der Clients am nächsten steht, als Splitter. Durch diese Art des Datenaustausches wird die zur Verfügung stehende Bandbreite am sparsamsten ausgenutzt.

Da die ständige Nachfrage der Multicast-Router nach interessierten Clients ebenfalls eine gewisse Netzlast verursacht, werden die Multicast-Signal-Pakete zusätzlich noch mit einer

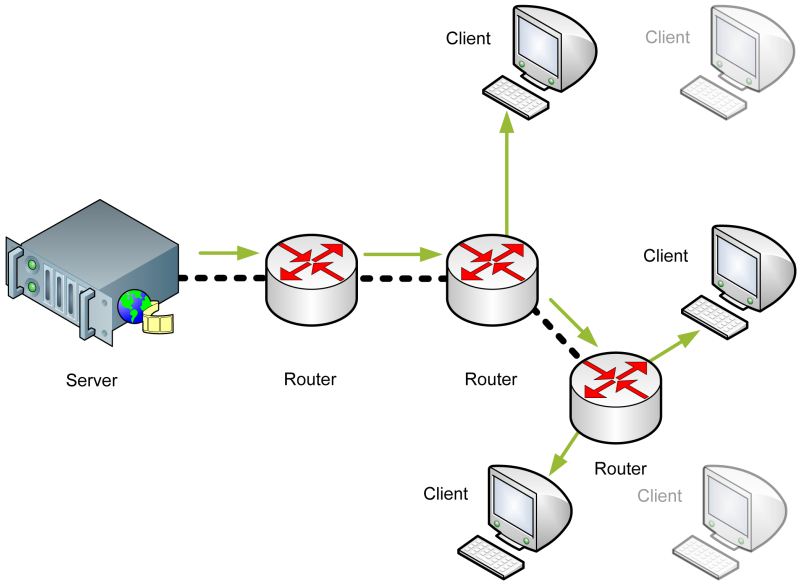


Abbildung 3: Darstellung des Multicastverfahrens

Lebensdauer versehen. Die Lebensdauer (Time-To-Live, oder TTL) ist eine Zahl, welche die Anzahl der Verbindungen zwischen den Multicast-Routern widerspiegelt. Bekommt ein Multicastsignal eine TTL von 3, so kann es nur 3 Router passieren. Danach wird es nicht mehr weitergeleitet. Mit diesem Eintrag, lässt sich die Ausstrahlweite eines Streamingangebotes begrenzen und somit auch die Netzlast erheblich verringern.(WEGNER R. : S.94)

2.3.3 Broadcast Strategie

Ein Broadcast beschreibt in der Informationstechnik einen Rundruf innerhalb eines Netzwerkes. Genauer ist es eine ungezielte Verbreitung von Informationen innerhalb einer Fläche. Typischerweise werden bei einem Broadcast die Datenpakete von einem Punkt aus an alle Teilnehmer übertragen. Ein klassisches Beispiel für eine Broadcast Verbreitung ist die Ausstrahlung terrestrischer elektromagnetischer Wellen von Langwelle bis UHF, von einem zentralen Sender an viele Empfänger.

Bei einem Broadcast haben die Empfänger keinerlei Einfluss auf die Informationen und auf die Präsentationen des Angebotes welches durch den Sender zur Verfügung gestellt wird. Die Empfänger können nur das empfangen, was gerade an Information ausgestrahlt wird.

2.4 Übertragungstechniken beim Streaming

Im Übertragungsmedium Internet ist der Benutzer zum größten Teil darauf angewiesen, seine von ihm gewünschten Informationen zu suchen, zu finden und schließlich abzurufen. Bei dem Streamen von Audio- und Videoinhalten wird neben diesem klassischen Verfahren,

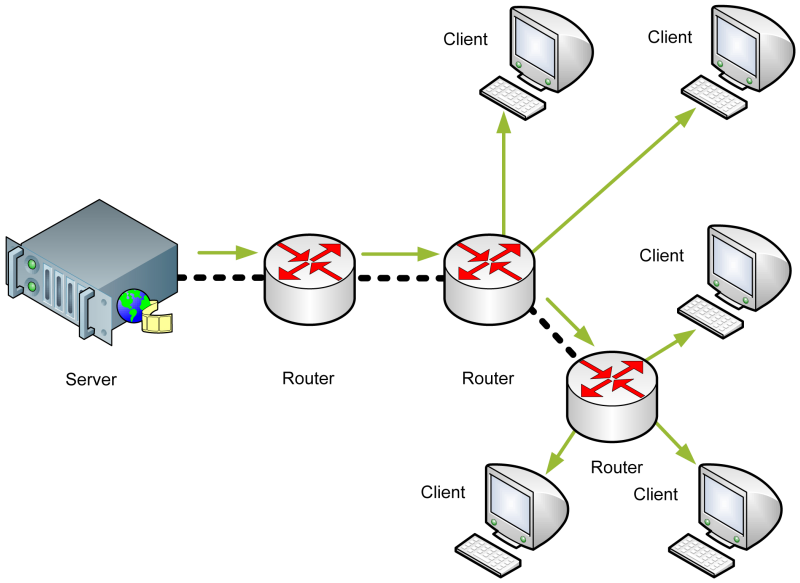


Abbildung 4: Darstellung des Broadcastverfahrens

welches als Pull-Verfahren bezeichnet wird, auch ein gegenteiliges Verfahren, welches als Push-Verfahren bezeichnet wird, genutzt.

2.5 Pull- und Push Verfahren

Bei dem sogenannten Push-Verfahren definiert der Benutzer im Voraus seine Interessengebiete oder seine gewünschten Programmsparten und lässt sich so die passenden Programmbeiträge und Informationen zusenden.

Mit diesem Verfahren wird versucht die Vorteile des klassischen Rundfunks und die Vorteile eines abrufbaren Archivs zu kombinieren. So erhält ein Benutzer zum Beispiel auch Inhalte und Informationen die in seinem Interessensspektrum liegen, von denen er jedoch keine Kenntnis hat, ihn aber dennoch interessieren könnten. So erhält ein Benutzer professionell zusammengestellte und ausgewogene Sendungen, wie er es vom klassischen Rundfunk gewohnt ist und hat dennoch Zugriff auf individuell von ihm zusammengestellte und zeitlich unabhängige Informationen.

Das Push-Verfahren lässt sich am Besten am Beispiel einer Webcam verdeutlichen. Hier lädt der Client einmal eine Befehlsroutine, die einen permanenten Download der Bildinformationen von der Webcam bewirkt. Danach verhält sich der Client selber passiv, bis er einen Befehl sendet um diese Verbindung abzubrechen.

Auch in der Streaming-Servertechnik spielen diese beiden Begriffe eine große Rolle. Bei einem Pull wird von einem Client oder einem Streaming-Server eine Aufforderung an einen Encoder gesendet um einen Datenstrom zu erhalten. Bei einem Push wird einmalig ei-

ne Verbindung von einem Encoder zu einem Streaming-Server aufgebaut, der die Daten anschließend verteilt.

2.6 Protokolle beim Streaming

Das Übertragen von Audio- und Videodaten über das Internet bedient sich einer Vielzahl an Protokollen und Mechanismen. Während statische Audio- und Videodaten auch über einen normalen Download, also unter zu Hilfenahme der Standard Internet-Protokolle HTTP und FTP, zum Benutzer übertragen werden können, benötigen Echtzeit-Inhalte einen anderen, komplexeren Weg. Da Echtzeit-Inhalte innerhalb einer kurzen Zeitspanne verschickt und empfangen werden müssen, bedarf es verschiedener Protokolle die diesen Datenverkehr durchführen, steuern und kontrollieren können.(BLACK U.)(LÖTZSCH S.)

2.6.1 Real Time Protocol (RTP)

Das RTP-Protokoll wurde speziell zur Unterstützung von Echtzeitübertragungen entworfen. Bei solchen Übertragungen reicht es in der Regel nicht aus, wie bei einer herkömmlichen Übertragung, eine Verbindung zwischen einem Client und einem Server aufzubauen, Gegeben falls benötigte Ressourcen zu reservieren und die Daten zu verschicken.

Für einen Echtzeit-Transport muss der Datenstrom in Pakete aufgeteilt werden. Diese Pakete werden getrennt voneinander an einen oder an mehrere Empfänger verschickt. Der oder die Empfänger muss/müssen diese Pakete empfangen und in der korrekten Reihenfolge wieder zusammensetzen und Gegeben falls mit anderen Signalen synchronisieren. Dabei stellt das RTP-Protokoll Ende-zu-Ende-Auslieferungsdienste für Daten mit Echtzeitcharakteristika zur Verfügung. Diese Dienste schließen folgende Leistungen ein: Identifizierung des Datentyps, Folgenummerierung, Zeitstempel und Auslieferungsüberwachung. Dabei unterstützt das RTP-Protokoll den Datentransfer zu mehreren Zielen und verwendet die Netzwerkstrategie Multicast, wenn diese vom zugrundeliegenden Netzwerk bereitgestellt wird.

Das RTP-Protokoll setzt im OSI-7-Schichtenmodell auf der Applikationsebene an, es ist also kein Teil der Transportschicht wie man vermuten könnte. Das Protokoll ist unabhängig von der zugrunde liegenden Transportplattform, wird aber meist zusammen mit den Protokollen IP und UDP verwendet. Da RTP selber keine Mechanismen bietet, um eine pünktliche Auslieferung oder andere Servicequalitäten zu gewährleisten, greift es auf die Dienste der unteren Schichten zurück.

Das RTP-Protokoll ist ein sogenanntes Kapselungsprotokoll. Das bedeutet, dass in dem Datenfeld des RTP-Paketes die zu transportierenden Echzeitinformationen stehen und die Informationen über den Verkehrstyp, also welche Daten das Paket gerade transportiert, im RTP-Header hinterlegt sind.

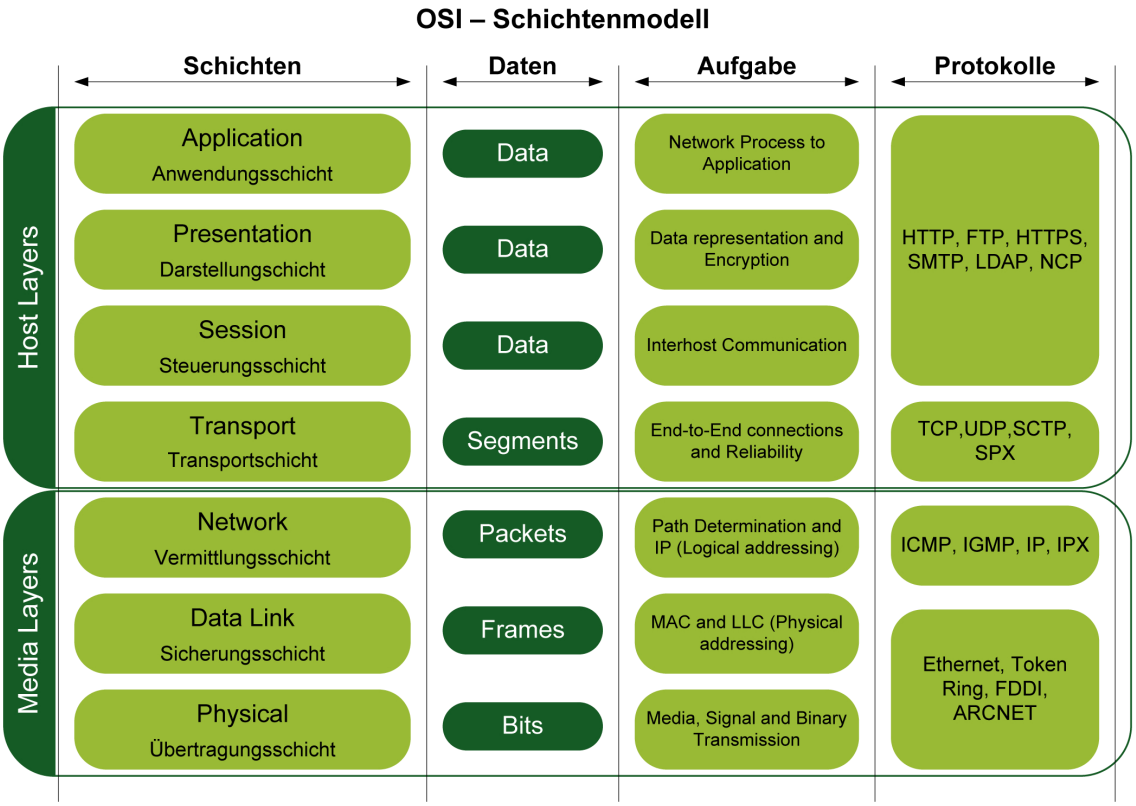


Abbildung 5: OSI-Referenzmodell

2.6.2 Real Time Control Protocol (RTCP)

Das Real Time Control Protocol (RTCP) ist für das Management zwischen Absender- und Empfängeranwendung während einer Echtzeitübertragung verantwortlich. Mit Hilfe dieses Protokolls kann der Absender den Empfänger über den RTP-Verkehr, den er empfangen sollte, informieren. Der Empfänger ist in der Lage Berichte über den Datenverkehr an den Sender zurückzuschicken. Der RTP-Verkehr findet also nur vom Sender zum Empfänger statt, während der RTCP-Verkehr in beide Richtung fließen kann.

Die Kontrolle des RTP-Verkehrs anhand des RTCP hat sich beim IP-Multicast als ziemlich nützlich erwiesen, da so leichter Fehler in der Paketverteilung festgestellt werden können.

Um den gesamten Verkehrsprozess der Absender- und Empfängerberichte während einer Sitzung konstant zu halten, egal wie viele Benutzer an dieser Sitzung teilnehmen, enthält das RTCP Funktionen um die Frequenz, mit der die Berichte versendet werden, festzulegen. Diese Berichte werden als Rundsendung an alle Teilnehmer geschickt; dadurch kann sich jede Anwendung über die Anzahl der verbreiteten Berichte auf dem laufenden halten und das Intervall der Berichterstattung entsprechend erhöhen oder reduzieren.

Um eine eventuelle Belastungsspitze, wenn zum Beispiel alle Benutzer zur gleichen Zeit einen Bericht senden würden, zu vermeiden, besitzt das RTCP Prozeduren zur Berechnung der Berichtsintervalle der einzelnen Benutzer innerhalb einer Sitzung.(BLACK U. : S.305/S.306)

2.6.3 Realtime Streaming Protocol (RTSP)

Das Realtime Streaming Protocol ist unter der Federführung von RealNetworks, Netscape Communications Corporation und der Columbia University entstanden, wurde durch den deutschen Informatiker Henning Schulzrinne (Foto rechts) spezifiziert und unter dem RFC 2326 zum Standard. Das RTSP ist ein Protokoll der Anwendungsschicht im OSI-Referenzmodell und ist für die Steuerung eines Echtzeit-Datenstrom konzipiert. Es kann sowohl unter der Verwendung verbindungsloser Protokolle (z.B.: UDP oder RDP), sowie unter der Verwendung von verbindungsorientierten Protokollen eingesetzt werden.

Es ergänzt somit die beiden Protokolle RTP und RTCP um die Möglichkeit die Inhalte der Multimedia-Streams über das IP-Netzwerk effizient übertragen zu können. Stark an das HTTP angelehnt, bietet es so die Möglichkeit Live-Daten oder Aufzeichnungen „fernzusteuern“ (z.B. „Play“, „Pause“, „Vorlauf“, „Rücklauf“, „Stop“) zu können.

Jeder Echtzeit-Datenstrom erhält, ganz ähnlich wie bei HTTP, eine eindeutige Netzadresse in der Form „rtsp://media.example.com:554/audio“. In dieser Netzadresse kennzeichnet „rtsp“, dass die RTSP-Kommandos über das Protokoll TCP (verbindungsorientiert) ausgetauscht werden sollen. „media.example.com“ ist hierbei ein beispielhafter DNS-konformer Host-Name und „554“ ist die Standard-Portnummer für RTSP, wobei auch andere Portnummern möglich sind.

Im Gegensatz zu HTTP, bei dem ein Client einen Server fragt und der Server dem Klienten antwortet, ist RTSP aber kein asymmetrisches Protokoll. Während der Client einen Media-Datenstrom abfragen kann, kann der Server eigeninitiativ bestimmte Abspielparameter abfragen.

RTSP unterstützt „unicast“ (Punkt-zu-Punkt Übertragung, z.B. für Video-on-Demand) und „multicast“ (Punkt-zu-Mehrpunkt Übertragung, z.B. für Video-On-Demand).(WEGNER R. : S.82)(LÖTZSCH S. : S.7)

2.6.4 Realtime Messaging Protocol (RTMP)

Das proprietäre Realtime Messaging Protocol (RTMP) wurde speziell von Adobe Systems entwickelt und dient dazu um Audio- und Videodaten in Echtzeit über das Internet von

einem Media-Server an einen Flash Player zu übertragen. Um eine gleichmäßige Nutzung der Daten zu gewährleiten, werden die Daten in größeren Datenblöcken übertragen.

Um Blockaden durch Firewalls entgegenzuwirken, kann dieses Protokoll unter Zuhilfenahme von Realtime Messaging Protocol Tunneled (RTMPT) in HTTP-Anfragen gekapselt werden. Eine verschlüsselte Verbindung mit SSL ist ebenfalls möglich. (ADOBE ONLINE 1)

2.6.5 Microsoft Media Server Protocol (MMS-P)

Das Streaming Protokoll Microsoft Media Server Protocol (MMS-P) wurde von Microsoft speziell für die zugehörige Server Software „Microsoft Windows Media Server“ entwickelt, welche ein Teil des Softwarepakets „Microsoft Media Server“ ist. Dabei ist das MMS-P ein proprietäres Format von Microsoft, verwendet eine ähnliche Technologie wie das RTP und setzt somit ebenfalls auf der Anwendungsschicht des OSI-Referenzmodells auf. Als Anwender-Software kommt typischerweise der „Windows Media Player“ zum Einsatz.

Das MMS-Protokoll wird von einem Client gestartet, der eine URL mit dem Präfix „mms://“ verarbeiten will. Er baut zunächst eine TCP-Verbindung zu dem Port 1755 des Servers auf, um seine IP-Adresse und einen selbst gewählten UDP-Port zu übermitteln. Daraufhin erzeugt der Server einen UDP-Socket und verbindet ihn mit dem gewünschten Port des Clients. Die Übertragung der Multimediadaten erfolgt dann über diese UDP-Verbindung, während die TCP-Verbindung für Steuerungsbefehle genutzt wird.

Es kann jedoch vorkommen, dass die UDP-Verbindung von einer Firewall verhindert wird. In diesem Fall kann der Client die Übertragung der Multimediadaten ebenfalls über die für diesen Zweck jedoch weniger gut geeignete TCP-Verbindung anfordern (MMST). Wenn auch das fehlschlägt, muss die dritte Möglichkeit genutzt werden, bei der die Kommunikation über HTTP erfolgt. (MICROSOFT ONLINE 1)

2.7 Streaming Lösungen

Basierend auf der Idee mediale Inhalte über das Medium Internet zu verbreiten, entwickelte sich das Thema Streaming schon am Anfang seiner Entstehung in verschiedene Richtungen. Während Microsoft in der Versuchung war, ein in sich geschlossenes Microsoft Netzwerk (MSN) zu entwickeln, ging die unter dem ehemaligen Microsoft Manager Rob Glaser neu gegründete Firma „Progressive Networks“ einen anderen Weg und verfolgte das Ziel die Client-Software zu verschenken, wie es auch schon bei der Profi-Browsersoftware Netscape erstmalig erfolgreich funktioniert hatte.

2.7.1 Entstehung der ersten Streaming Lösungen

1994 erschien der erste RealAudio Player 1.0 als frei downloadbare Software. Das Business-Modell sah vor, ähnlich wie bei dem Erfolg des Netscape-Browsers, dass der Player kostenfrei bleiben sollte, jedoch die Server, die den Stream erzeugen, zu bezahlen sind. Am Ende der Betatestperiode, im Juli des Jahres 1995 waren bereits mehr als zweihunderttausend Decoder heruntergeladen worden, diese Zahl sollte sich in den folgenden drei Monaten sogar noch verdreifachen. Obwohl die Geschwindigkeit des zur Verfügung stehenden Netzes damals gerade einmal 14,1 Kbps betrug und somit nur Audio allein als Information in Frage kam.

Die encodierten Daten, die in dem Real-Media-File-Format von RealNetworks abgespeichert wurden, ermöglichten es die Inhalte ab einer beliebigen Startposition bis zur gewählten Endposition abzuspielen. Dadurch war es erstmalig möglich Radiosendungen nicht physikalisch von einander zu trennen um einzelne Beiträge einer Sendung abrufbar zu machen.

Die damaligen Player und Server von RealNetworks boten schon wie die modernen Player heutzutage die Möglichkeit verschiedene Transportprotokolle für die Übertragung der Daten zu verwenden. Diese Auswahl erfolgte entweder durch den Benutzer über seinen Player, oder automatisch vor der Übertragung durch den Player selbst. Dazu fähig sind die Player, indem sie zuvor testen ob der Datenstrom über Multicast angeboten wird. Ist eine Übertragung über Multicast nicht möglich, versucht ein Player die Daten über eine Unicastverbindung und UDP anzufordern. Schlägt das ebenfalls fehl, weil sich der Rechner des Clients zum Beispiel hinter einer Firewall befindet, versucht ein Player eine Verbindung über TCP oder schließlich über HTTP herzustellen.

Um die bestmögliche Übertragung der Daten zu dem Benutzer gewährleisten zu können, verwendete RealNetworks die sogenannte „Bandwidth Negotiation“. Dabei erhielt der Server vor dem Start der Übertragung eine vom Benutzer im Player voreingestellte verfügbare Bandbreite, anhand er entscheiden konnte, welche der bandbreitenoptimierten Daten er für diesen Nutzer zur Verfügung stellt.

Im Laufe der nächsten 3 Jahre erschienen 4 weitere Releases des RealPlayers, der jetzt unter dem Firmennamen RealNetworks und in Kooperation mit Microsoft vertrieben wurde und auch die Erweiterung des Videoaustausches bot. Diese Kooperation brach jedoch schon einige Zeit später auseinander, als sich der Markt für Echtzeit-Multimedia konsolidierte und ehemalige Konkurrenten wegbrachen oder aufgekauft wurden. So entwickelten sich die beiden ehemaligen Partner RealNetworks und Microsoft zu ihren jeweils stärksten Rivalen.

Der erste Live-Encoder wurde im September 1995, anlässlich eines Basketballspiels vor-

geführt. Wenig später hielt das Live-Encoding und damit das erste Live-Streaming bei dem amerikanischen Sender ABC Einzug um Nachrichtensendungen im Web auszustrahlen. Eine der ersten Ausstrahlung war dabei die Gerichtsverhandlung gegen O.J. Simpson.(WEGNER R. : S. 44/45)(Fischer H. : S.8)

Angesichts dieser Entwicklung, kann man die Firmen Microsoft und RealNetworks getrost als Urväter des Echtzeittransportes von Audio- und Videodaten im Internet bezeichnen. Heutzutage gibt es wesentlich mehr Anwärter um die ersten Plätze dieses Marktes. Durch die ständig wachsende Datenübertragungsrate und den damit verbundenen Möglichkeiten im Internet, sowie die Vereinigung oder den Aufkauf von kleineren Firmen, hat das Thema Streaming in seiner technischen Entwicklung eine Halbwertszeit von unter einem halben Jahr.

Mit dem MP3-Format entwickelten sich gerade im Audiobereich Alternativen nicht nur für die Player, sondern, noch viel wichtiger, kostenlose Alternativen zu den Servern, die die Stream bereitstellen können. So ist es heutzutage möglich einen Stream nicht nur mit kostenloser Software zu hören, sondern ebenfalls einen Stream mit kostenfreier Software für das Internet bereitzustellen.

Ebenso entwickelte sich mit der Verfeinerung der Entwicklungsumgebung Adobe Flash (anfänglich Macromedia Flash), die die Möglichkeit der Verbreitung multimedialer Inhalte als Flash-Filme bietet, das Thema Streaming in eine mit noch mehr Möglichkeiten gespickte Lösung. Mittels Flash liese sich zum Beispiel der komplette Inhalt einer Nachrichtenorganisation multimedial aufbereitet auf einer Website darstellen. Egal ob Videos, Bilder, Texte oder das gesprochene Wort.

Der Trend, Musik über das Internet zu hören, geht aber einen anderen noch interessanteren Weg. Immer mehr entwickelt sich die Möglichkeit die Streams über mobile Endgeräten zu hören, sei es ein Handy oder ein speziell für diesen Weg entwickeltes „Internetradio“. So ist es zum Beispiel möglich in der Küche ohne Laptop oder PC seinem Lieblings-Stream aus dem Internet zu lauschen.

2.7.2 Realmedia

Ein möglicher Workaround um Audio- und Videodaten über das Internet zu streamen, ist die Verwendung des Streaming-Server „Helix Server“ des US-amerikanischen Unternehmens Real Networks und des hauseigenen Media Player „RealPlayer“. Der Helix Server ist in der Lage Real Media, Windows Media, QuickTime, MP3, H.264, AAC Inhalte verteilen zu können. Im Gegenzug zu dem kommerziellen Helix Server, gibt es eine nicht-kommerzielle Version, den Helix DNA Server, dessen Quellen als Open-Source zur Verfügung stehen. Dieser Server ist jedoch nur

in der Lage MP3, RealAudio- und RealVideo-Dateien und durch einen Patch OGG/Vorbis zu verteilen.

Der Helix Server ist für die Betriebssysteme Linux, Windows 2003 Server und Solaris 10 verfügbar und in der Lage sowohl für verdrahtete und drahtlose Geräte Multimediainhalte bereitzustellen.

Als Player für die hauseigenen Formate RealAudio und RealVideo bietet RealNetworks den RealPlayer, der in einem späteren Kapitel noch einmal ausführlicher vorgestellt wird, an. (REAL ONLINE 1)

2.7.3 Windows Media

Ein weiterer möglicher Workaround ist die Verwendung des von Microsoft entwickelten Media Server, welcher über das MMS-Protokoll Audio- und Videodaten zu Verfügung stellen kann. Dabei ist diese Media Serversoftware ein zusätzlich angebotener Dienst der bei einem Windows Server 2003 aktiviert werden kann. Mit diesen Erweiterungen ist es möglich On-Demand und Liveinhalte über den Windows Server zu verteilen.

Als Live-Encoder kommt dabei der Windows Media Encoder zum Einsatz, für die Bereitstellung von On-Demand Inhalten werden die Dateien lediglich in einem Verzeichnis (%systemdrive%) hinterlegt und sind im Anschluß unter der IP der Servers und unter Verwendung des MMS-Protokolles (mms://) direkt abrufbar. (MICROSOFT ONLINE 2)

2.7.4 Flash Streaming Lösungen

Mit dem Vormarsch der animierten Darstellung von Bildern, Texten und Sounds anhand von Flash Filmen im Internet, waren auch die Entwickler der Browser gezwungen ihren Produkten die Möglichkeit zu geben diese Filme darstellen und abspielen zu können. Entstanden ist dadurch auch ein für das Streaming interessanter Weg der Verteilung, denn es lassen sich so über einen Flash Player nahezu alle Browser und somit auch viele Betriebssysteme erreichen.

Neben der Möglichkeit auf einem Server gespeicherte Audiodateien über einen Flash Film zu verbreiten, bietet dieser Weg zusätzlich auch die Fähigkeit Videos abzuspielen. Unter Zuhilfenahme von Flash Media Servern lassen sich ebenfalls Live- Audio und Videoinhalte mit einem Flash Player nutzen. Somit ist eine Streaminglösung mittels Flash ein Allroundtalent, wenn es um das plattformübergreifende Verteilen von On-Demand und Liveinhalten geht.

Auch bietet dieser Weg der Verteilung eine einfache Möglichkeit Zusatzinformationen zu den Streams bereitzustellen. Informationen wie der Titel des aktuellen Liedes, aktuelle Nachrichten in einem Ticker oder ein Kaufangebot für die CD von dem eben gehörten Künstler, lassen sich in so eine Streaming Lösung einbauen.

Streaming-Server, die einen solchen Workaround zur Verfügung stellen können, sind der Adobe Flash Media Server oder der Wowza Media Server. Diese beiden Streaming-Server werden in einem nachfolgendem Kapital noch einmal genauer erläutert und beschrieben.

2.8 Streaming Formate

Für das Streamen von Audio- und Video Daten wurden verschiedene Dateiformate entwickelt. Diese Formate werden auch als Containerformate bezeichnet, welche verschiedene andere Datenformate enthalten und transportieren können. Dabei gibt das Containerformat an, in welcher Art und Struktur der Inhalt aufzubewahren ist. Im Falle von Videodaten ist somit ein Container nötig, der zumindest Audio- und Videodaten beinhalten kann.

Anfänglich waren diese Formate proprietär, später kamen dann einige freie Formate nach. Die meisten proprietären Player unterstützen standardmäßig nur jeweils die hauseigenen Streaming-Formate, jedoch gibt es die Möglichkeit per Updates die Player um eine Vielzahl anderer Formate zu erweitern.

Abbildung 6: Auszug einiger Containerformate die in der Lage sind Audio- und Videoinhalte zu transportieren.

2.9 Streaming-Server

Eine der wichtigsten Komponenten bei dem Streamen von Audio- oder Video Daten, ist die Komponente, die die aufbereiteten Inhalte dem Benutzer zur Verfügung stellt. Der sogenannte Streaming-Server.

Streaming-Server haben die Aufgaben die Streams zu verwalten, an die auf ihn verbundenen Klienten weiterzuleiten und zusätzliche Informationen über die Streams bereitzustellen. Dabei ist ein Streaming-Server nicht als reine Hardwarekomponente zu verstehen, sondern vielmehr als Software, die auf einem Rechner läuft. So können sich Streaming-Server die MP3-Streams und Streaming-Server die Flash Streams erstellen, verwalten und verteilen physikalisch auf einem Rechner befinden.

Im Bereich des Audio-Streaming gibt es bereits eine Vielzahl von Streaming-Server, die ihre Inhalte an Media Player, Flash Player oder mobilen Endgeräten ausliefern können. Gleichzeitig gibt es auch kostenfreie Server, die keinen Lizenzen unterliegen.

Im Bereich des Video-Streaming ist die Auswahl an Server Software etwas begrenzter, was unter anderem dazu führt, dass keine kostenfreie Produkte existieren.

Aufgaben eines Streaming-Servers

- Streaming-Server verwalten Streams.

Ein Streaming-Server nimmt aufbereitete Streams in Empfang und teilt ihnen Namen, Beschreibungen und andere Eigenschaften zu. Bei einem Ausfall oder einer Störung kann der Server mit einer Umleitung auf einen anderen Stream reagieren.

- Streaming-Server verteilen Streams.

Benutzer können sich auf einen Streaming-Server verbinden, zum Beispiel auf einen Mountpoint, und den dort hinterlegten Stream nutzen.

- Streaming-Server stellen Zusatzinformationen (Metadaten) zur Verfügung.

Ein Streaming-Server ist in der Lage Zusatzinformationen innerhalb der Streams zur Verfügung zu stellen. So erhalten die Benutzer innerhalb ihrer Media Player Informationen zu dem aktuellen Inhalt.

In den folgenden Kapiteln werden zwei Streaming-Server, die durch ihren Funktionsumfang für eine geeignete Streaminglösung in Frage kommen, vorgestellt.

2.9.1 Icecast Streaming-Server

Ein bekannter Audio-Streaming-Server nennt sich Icecast. Er eignet sich unter anderem für das Streamen von MP3, Ogg und AAC-Inhalten. Dieser Streaming-Server ist ein Open-Source Projekt und für diverse Linux und Windows-Systeme verfügbar. Durch seine vielfältigen Einsatz- und Streamingmöglichkeiten, ist dieser Server auch bei den Amateuren und bei Semiprofessionellen Streaming Anbietern sehr beliebt. Der Icecast Streaming-Server arbeitet mit sogenannten Mountpoints. Diese ermöglichen es ihm mehrere Streams über die selbe IP-Adresse und den selben Port zur Verfügung zu stellen. Diese Streams können dann über den Namen dieses Mountpoints direkt abgehört werden.

Beispiele für eine Adressierung anhand von Mountpoints

```
http://Icecast-IP-Adresse:Icecast-Port/Mountpoint
```

Aufbau einer Streamadresse über die ein Stream mittels der Angabe eines Mountpoints von einem Icecast Server abgerufen werden kann.

```
http://192.168.0.3:8000/stream1
```

Hier wird ein Stream, der über den Mountpoint „stream1“ erreichbar ist, über den Port „8000“ an der IP-Adresse „192.168.0.3“ angesprochen.

```
http://192.168.0.3:8001/stream2
```

In diesem Beispiel wird ein Stream, der über den Mountpoint „stream2“ zu erreichen ist, über den Port „8001“ an der IP-Adresse „192.168.0.3“ angesprochen.

Der Port über den der Icecast Server sein Streamingangebot zur Verfügung stellt lässt sich beliebig anpassen, der Standardport ist jedoch der Port 8000. Die meisten Icecast Server senden über den Port 8000 oder/und 8001 ihre Inhalte. Konfiguriert wird der Icecast Server etwas umständlich von Hand über eine zentrale XML-Datei.(ICECAST ONLINE 1)

Auf die Installation und Konfiguration eines solchen Streaming-Server, wird in einem nachfolgendem Kapitel noch ausführlicher eingegangen.

2.9.2 Wowza Media Server

Der Wowza Media Server, aus dem Hause „Wowza Media Systems“, ist eine auf der Programmiersprache Java basierender und somit eine vom zugrundeliegenden Betriebssystem unabhängige Serverapplikation. Sie bietet eine skalierbare und höchst effiziente Plattform zur Bereitstellung von Live- oder On-Demand Audio- und Videoinhalten in dem weit verbreiteten Flash Format.

Verfügbar ist der Wowza Server für die Betriebssysteme Solaris, Microsoft Windows (XP, Vista, 7, Server 2003 & 2008), Mac OS X, Unix und für Linux (alle Distributionen). Dieser Media Server kann über die Streaming-Protokolle RTP/RTSP und RTMP Inhalte ausliefern und ist ebenfalls in der Lage hochqualitative Videos im H.264/HE-AAC Format zu verschicken.

Der Wowza Media Server bietet sich sehr für das Umwandeln von vorhandenen MP3-Streams in Flash-Streams an. Er kann zum Beispiel die MP3-Streams von einem Icecast Server aufgreifen, umwandeln und diese anschließend als Flash Streams bereitstellen. Der Benutzer ist dadurch in der Lage über einen Flash Player, der zum Beispiel auf einer Website eingebunden ist, diese Inhalte abzurufen. Bei der Umwandlung der MP3-Streams in ein Flash konformes Format gehen die Metadaten nicht verloren, somit ist es auch möglich mit dem Wowza Media Server die Metadaten in nahezu Echtzeit zu den Flash Player zu übertragen. Die Nutzung eines Wowza Servers hat den großen Vorteil, dass das Mitschneiden oder der Download der Streaming-Inhalte nahezu komplett unterbunden werden kann.

Ebenfalls ist der Server in der Lage, ein Live Video Signal von einem Video Live Encoder, zum Beispiel dem Adobe Flash Media Encoder, empfangen zu können und an einen Flash Player weiterzuschicken. Durch seine integrierte On-Demand Funktionalität ist er ebenfalls dazu fähig, Video- und Audiosequenzen innerhalb eines Flash Players wiederzugeben.

Für den Wowza Media Streaming Server gibt es eine kostenlose Testversion zum Download auf der Webseite des Herstellers, mit dieser Testversion können jedoch nur maximal 10 Streams gleichzeitig versendet werden. Nach der Installation des Wowza Server, findet man sehr gut erklärte Beispiele (inklusive der Serverseitigen Java - und der Klientseitigen Actionskript Quelltexte) im Installationsordner vor. (WOWZA ONLINE 1)

2.10 Kodierung und Kompression

Ein wichtiger Arbeitsschritt bei der Übertragung von Audio- und Video Daten über das Internet, ist die Wahl des Formats und in welcher Qualität diese Daten ausgeliefert werden sollen. Bedingt durch die Bandbreitenbegrenzung, die jede Internetverbindung mit sich bringt, ist es unerlässlich zum Beispiel ein Video in seiner Qualität und Auflösung zu mindern und somit erst einmal das Streaming überhaupt zu ermöglichen. Ebenso müssen die zu streamenden Daten an die Plattform, die die zu erwartenden Endbenutzer bevorzugen, angepasst werden. So macht es einen großen Unterschied in der Kodierung und Kompression ob man einen Video-Stream an ein mobiles Endgerät, oder eher einen Audio-Stream an einen Media Player wie den Winamp senden möchte.

Um einen Echtzeitstrom bestmöglich in das Internet und eventuell zu einem bestimmten Endgerät versenden zu können, benutzen Streaming-Systeme Kodierungs- und Kompressionsalgorithmen um die Inhalte an die Gegebenheiten und Bedürfnisse anzupassen.

Der Begriff „Kodierung“

Die Umsetzung einer Information auf eine bestimmte Übertragungstechnik, zum Beispiel

das Versenden einer Nachricht mittels dem Morsealphabet, wird als Kodierung bezeichnet. So ist es möglich Informationen über eine weite Strecke zu übertragen, jedoch können nur der Sender (Encoder) und der Empfänger (Decoder) anhand der Verschlüsselung (der Code) die Nachricht lesen.

Der Begriff „Komprimierung“

Bei der Kompression von Daten werden unwichtige oder doppelte Informationen aus der zu übertragenden Nachricht entfernt um somit die zu übertragende Datenmenge reduzieren zu können.

Ein klassisches Beispiel für eine Kompression ist die Versendung einer Nachricht über ein Telegramm. Hierbei werden einfach unwichtige Wörter weggelassen, jedoch bleibt die eigentlich Information für den Empfänger sehr gut erhalten. Eine redundante, also überflüssige Information, wäre zum Beispiel die Angabe. „Null Uhr morgens“. In einem Telegramm würde die überflüssige Information „morgens“ einfach entfernt, da „Null Uhr“ nur am Morgen sein kann.

Bei der Komprimierung von Audio-Daten werden zum Beispiel kaum hörbare Töne entfernt, bei der Komprimierung von Videodaten verzichtet man beispielsweise auf Teile der Farbinformationen oder gleiche Bilder / Bildfolgen.

Während also die Kodierung von Daten einen Wechsel in ein effizienteres digitales Basissystem darstellt, wobei die Eingangsdaten unverfälscht bleiben, sodass sie jederzeit wieder vollständig hergestellt werden könnten, beschreibt die Kompression das Eliminieren von redundanten und wenig relevanten Eingangsdaten, die im Anschluss nicht wieder hergestellt werden können. (WEGNER R. : S.28/S.29)

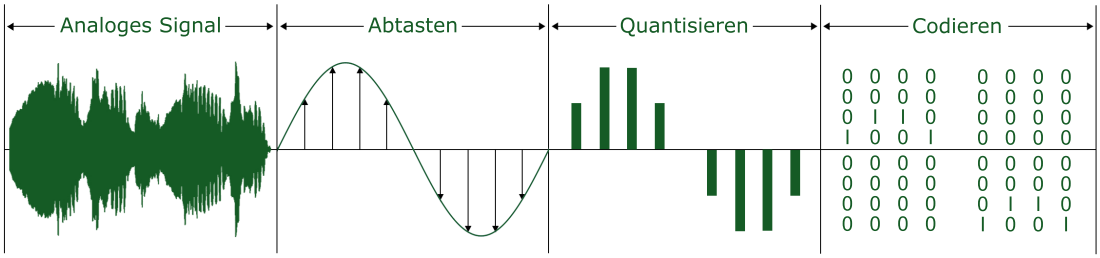


Abbildung 7: Beispielhafte Darstellung eines Encodingprozesses

2.10.1 Der Begriff „Encoding“ beim Streaming

Wenn man bei dem Thema „Streaming“ von Encoding (oder Enkodierung) spricht, dann meint man im Allgemeinen das gleichzeitige Kodieren und Komprimieren eines Signals in ein für das Streaming erforderliches Format und Qualität. Also zum Beispiel das Wan-

deln eines Signals aus dem Audioausgang einer Soundkarte in ein digitales und qualitativ vermindertes Audiosignal, welches dann mit Hilfe eines Streaming-Servers verteilt werden kann.

2.10.2 Der Begriff „Encoder“ und seine Aufgaben

Der Encoder ist bei dem Streamen von Audio- und Videodaten diejenige Arbeitseinheit die bestimmt in welcher Qualität und in welchem Format der letztendliche Stream zum Einsatz kommt.

Mit Hilfe eines Encoder können jedoch noch zusätzliche Eigenschaften eines Streams definiert werden. Zum Beispiel die Anzahl der Kanäle (Mono oder Stereo) die im Stream übertragen werden sollen, die Bitrate (z.B.: 128kbps) und die Frequenz mit der das Ursprüngliche Signal abgetastet werden soll (Samplingfrequenz. z.B.: 44khz), sowie zusätzliche Text-Informationen im Header des Streams (z.B.: der Name des Streams).

Ein Encoder hat also innerhalb einer Streaminglösung nicht nur die Aufgabe das ursprüngliche Eingangssignal in das gewünschte digitale Ausgangssignal zu wandeln, sondern er regelt zusätzlich noch die Kompression und hilft somit die Qualität des Ausgangssignales zu steuern.

Ebenfalls hat ein Encoder die Aufgabe zu bestimmen, wohin und unter welchen Namen das produzierte Ausgangssignal geschickt werden soll. Somit meldet er sich, zum Beispiel an einem Streaming-Server, mit einem Benutzernamen und einem Passwort an und übergibt sein Signal an einen von ihm festgelegten Punkt, welcher auch als Mountpoint bezeichnet wird.

Da also ein Encoder innerhalb eines Streaming Systems mehrere Aufgaben und Bereiche abdeckt und somit seine eigentliche Aufgabe des reinen Kodierens überschreitet, werden diese Arbeitseinheiten auch als „Sourceclient“ bezeichnet.

2.10.3 Beispiele für Encoder und Sourceclients

Für das Encodieren und Komprimieren der Signale von Audio- und Videoquellen, ob in Echtzeit oder On-Demand, gibt es die verschiedensten Lösungen. Der größte Unterschied bei der Durchführung von diesem Arbeitsschritt, ist wohl die Unterscheidung zwischen der Nutzung von Hard- und Softwareware Encodern.

Für die Verarbeitung eines Audiosignals gibt es sehr leistungsfähige Hardware-Encoder, die in der Lage sind die meisten freien Streaming Formate zu erzeugen. Eine solche Hardwarelösung bildet eine sehr stabile und ausfallsichere Einheit innerhalb eines Streamingsytems.

In den folgenden Kapiteln werden einige Lösungen für Software-Encoder und eine Lösung für einen Hardware-Encoder vorgestellt und näher erläutert.

2.10.4 Hardware-Encoder

Die größten Vorteile eines auf Hardware basierenden Encoders, sind seine Ausfall- und Netzwerksicherheit sowie die Skalierbarkeit des entstehenden Systems. Im Gegensatz zu einem Software-Encoder läuft ein Hardware-Encoder nicht auf einem Rechner auf dem höchstwahrscheinlich noch andere Dienste und Programme laufen, sei es ein Streaming-Server oder ein Web-Server.

Eine fehlerhafte Administration oder Konfiguration eines auf einer Software basierenden Encoding Systems hat auch Auswirkungen auf das Encodieren und Bereitstellen des Streamingangebotes. Auch ist nach einem Totalausfall der Rechentechnik eine aufwändige und zeitraubende Wiederbeschaffung, Installation und Konfiguration des gesamten Systems notwendig. Sollte doch mal ein Hardware-Encoder ausfallen, kann sofort das defekte Gerät mit wenig Aufwand durch ein anderes ersetzt werden.

Auch sind einer Skalierung mittels Hardware-Encoder fast keine Grenzen gesetzt, werden mehr Encoder benötigt ist es relativ einfach diese zu erweitern, solange es der zur Verfügung stehende Platz, in zum Beispiel einem Serverraum, zulässt. Eine auf Software basierende Encoder Lösung stößt jedoch schnell an die Grenzen der vom Rechner bereitgestellten Ressourcen.

Ein nicht zu verachtender Nachteil einer solchen Lösung mittels Hardware-Encodern, sind die Kosten. Auch wenn das Preis-Leistungsverhältnis durchaus berechtigt und stimmig ist, ist es ein großer finanzieller Mehraufwand von Nöten um mehrere Streams mit verschiedenen Bitraten zu erzeugen.

Im Bereich des professionellen Streaming greift man lieber auf eine hardwarebasierenden Encoder Lösung zurück, im semiprofessionellem Bereich reicht es meist völlig aus auf Software Lösungen zu setzen.

Vor- und Nachteile einer Hardwarelösung

- − Teurer als eine Softwarelösung
- − Benötigt mehr Platz im Serverraum
- + Extrem ausfallsicher
- + Leicht administrierbar

+ Leicht auswechselbar

2.10.5 BARIX Hardware-Encoder

Ein Audio-Hardware-Encoder ist der INSTREAMER der Firma BARIX. Der INSTREAMER ist in der Lage von einer Audioquelle bis zu acht digitale Audioquellen (unterschiedliche Formate) gleichzeitig zur Verfügung zu stellen. Der Encoder kann problemlos in einem 19 Zoll Rack montiert werden, dabei passen bis zu vier dieser Encoder nebeneinander in einen Einschub.

Der BARIX INSTREAMER kann analoge- und digitale Audioquellen abhören und das kodierte Signal innerhalb eines Netzwerkes zur Verfügung stellen. Konfiguriert und administriert wird der Encoder standardmäßig über ein Web-Interface innerhalb des LAN/WAN.

2.10.6 Software-Encoder

Ein großer Nachteil beim Encodieren unter Zuhilfenahme eines Software-Encoders, ist die Anfälligkeit und Sicherheit des Systems. Für den privaten und semiprofessionellen Gebrauch ist diese Art der Encodierung, auf Grund der kostengünstigen Anschaffung, von Audio- und Video Daten wohl am Besten geeignet. Für das professionelle Streaming, bei dem die Themen Ausfall- und Netzwerksicherheit eine große Rolle spielen, eignet sich eine hardwarebasierende Lösung besser.

Einige Vorteile der softwareseitigen Encodierung, dürften neben dem Kostenfaktor auch der dynamische und mehrfache Einsatz der Software sowie der benötigte Platzbedarf sein. Eine Software muss nur installiert und konfiguriert werden, ein Hardware-Encoder braucht jedoch zusätzlich Platz.

Vor- und Nachteile einer Softwarelösung

+ Keine extra Hardware nötig
- Zeitaufwändiger Administrationsaufwand
- geringere Ausfallsicherheit

2.10.7 Ices, ein Sourceclient

Ein beliebter Client um das Encoding von Audio-Datenquellen zu organisieren und durchzuführen, ist Ices. Ices wird im Zusammenhang mit dem Icecast Server entwickelt und stellt somit einen idealen Sourceclient für diesen Server dar. Mit diesem Tool lassen sich lokale Musikdateien, die in

einer Playlist zusammengefasst werden, oder auch der Input eines Line-In einer Soundkarte an einen Streaming-Server übermitteln. Ices ist in der Lage die Formate MP3 (Ices 0.3) und Ogg Vorbis (Ices2) für einen Streaming-Server zur Verfügung zu stellen. Ices ist momentan für die Plattformen Linux (Redhat, Debian, Ubuntu), FreeBS, OpenBS, Solaris erhältlich.

Seit 2004 heißt Ices Ices2 und wartet mit dem Ogg/Vorbis Format auf. Allerdings ist in dieser Version 2 des Klienten das Streamen von MP3 Streams nicht mehr möglich, da das MP3 Format eine patentierte und proprietäre Kodierungstechnologie ist und laut den Entwicklern kein großes Interesse mehr an dieser Art von Kodierung besteht. Ogg/Vorbis dagegen ist eine patentfreie Audio-Kodierungstechnologie.

Es ist nicht möglich einen MP3-Live-Stream mit Ices zu erzeugen. Ices0 kann zwar einen MP3-Stream erzeugen, aber nur anhand von lokalen Musikdateien, die in einer Playlist zusammengefasst sind. Ices2 kann zwar einen Live-Stream erzeugen, indem er den Ausgang einer Soundkarte abgreift, aber nur im Ogg-Format weiterreichen. Eine Lösung für dieses Problem stellt DarkIce dar.

2.10.8 DarkIce, ein Sourceclient

DarkIce ist ein freier Live-Audio-Encoder der von verschiedenen Audiogeräten Inhalte aufnehmen, encodieren und an einen Streaming-Server weitergeben kann. Im Gegensatz zu Ices ist dieser Encoder in der Lage die Ausgabe einer Soundkarte in einen MP3-Stream zu wandeln und weiterzugeben. Dadurch ist es mit diesem Encoder möglich einen MP3-Live-Stream zu erzeugen und an einen Icecast Server weiterzuleiten.

Der Encoder läuft unter den Betriebssystemen FreeBSD, Linux, MacOS X, NetBSD / OpenBSD und SUN Solaris und kann seine Daten an verschiedene Streaming-Server wie Shoutcast, Icecast und den Darwin Streaming Server senden. Dabei unterstützt er unter Verwendung verschiedener Bibliotheken die Encodierung in die Formate MP3, MP2, Ogg/Vorbis, AAC und AAC HEv2.

2.10.9 Adobe Flash Media Live Encoder

Der Adobe Flash Media Live Encoder ermöglicht es einen Live-Stream für einen Flash Media Server zur Verfügung zu stellen. Dabei kann der Encoder von verschiedenen Echtzeitquellen wie zum Beispiel Webcams, Soundkarten oder anderen Plug-and-Play Kameras Video- und Audiosignale aufnehmen und an einen Streaming-Server weitersenden.

In seiner aktuellen Version 3 unterstützt dieser Live Encoder standardmäßig das Nellymoser- und MP3-Format. Mit einem entsprechenden Encoder-Plug-in können auch AAC und HE-

AAC verwendet werden.

2.11 Player Software und Endgeräte für das Streaming

Um Audio-Streams überhaupt empfangen und nutzen zu können, gibt es verschiedenste Möglichkeiten und Techniken. Der gängigste und am häufigsten verwendete Weg ist das Nutzen über eine Software auf dem heimischen PC oder auf einem Laptop. Je nach Betriebssystem gibt es diverse Media Player die bereits mit dem Betriebssystem installiert werden. Zusätzlich gibt es für die meisten Betriebssysteme diverse Media Player die nachträglich installiert und genutzt werden können.

In den folgenden Kapiteln, werden einige Beispiele von Media Playern aufgezählt und erläutert.

Microsoft Windows Media Player

Der Windows Media Player ist der Multimedia Player von Microsoft und aktuell in der Version 12 zu haben. Er lässt sich einfach über das Betriebssystem nachinstallieren und bietet so die Möglichkeiten alle persönlichen digitalen Medien zu verwalten und wiederzugeben. Ebenfalls ist dieser Player in der Lage Audio und Video-Streams aus dem Internet abzuspielen. Microsoft entwickelte für den Player eigene Formate, Microsoft Windows Media Video (WMV) für das wiedergeben von Video und das Microsoft Windows Audio (WMA) für die Wiedergabe von Audio. Diesen Media Player gibt es auch als Plugin für alle gängigen Browser. (MICROSOFT ONLINE 3)

Apple QuickTime Player

Der QuickTime Player von Apple ist ein kostenlos erhältlicher Media Player und ist in der Lage unterschiedliche Dateitypen abzuspielen. Der QuickTime Player ist für verschiedene Betriebssysteme, z.B. Mac OS oder Windows per Download verfüg- und installierbar. Ebenfalls ist der Player als Plugin für die gängigsten Browser verfügbar. Als eigenes Format bietet Apple zum Beispiel das QuickTime Movie.

RealNetworks RealPlayer

Den Media Player RealPlayer von der Firma RealNetworks gibt es für die Betriebssysteme Windows, Mac und Linux. Der Player unterstützt neben den hauseigenen Formaten RealAudio und RealVideo das Wiedergeben von MPEG-1 Layer (MP3), MPEG-4 und den Formaten von QuickTime. Der RealPlayer gilt als einer der ersten Media Player für das Hören von Streaming Media.

Nullsoft Winamp

Der Winamp Player ist ein von der Firma Nullsoft entwickelter Media Player für das Betriebssystem Windows und momentan in der Version 5.57 erhältlich. Dieser Player war einer der ersten Player der in der Lage war SHOUTcast-Streams abzuspielen. Heutzutage ist er nicht nur ein Player für Audio sondern eignet sich ebenfalls für das Wiedergeben von Video. Der Winamp Player ist in der Lage MP3-, MP4-, (Ogg)Vorbis-, AAC-,WMA-, WMV und viele andere Formate abzuspielen. (WIKI ONLINE 1)

NOXON Internetradio von TerraTec

Der NOXON von Terratec ist ein 1kg schweres und mit einem 5 Watt Lautsprecher versehenes Internetradio der Firma TerraTec. Der NOXON kann über WLAN (54 Mbit/s) oder einer RJ-45 Buchse (10/100 MBit/s LAN) die Musiksammlung vom heimischen PC abspielen und bietet unter zur Hilfenahme eines Routers und eines Internetanschlusses Zugriff auf viele Internetradios. Dabei unterstützt der NOXON Streams im MP3-Format, AAC+ und WMA-9 (bis zu 320 kBit/s). Mittels einer großen LCD-Anzeige auf der Vorderseite, stellt dieses Internetradio auch die Metadaten eines Streams als Laufschrift dar. (TERRATEC ONLINE 1)

2.12 Der Begriff „Metadaten“

Metadaten sind im Allgemeinen Daten die andere Daten beschreiben. Von Tim Berners-Lee, dem Erfinder des World Wide Web und Direktor des World Wide Web Consortiums (W3C), stammt die Definition: „Metadaten sind maschinenlesbare Informationen über elektronische Ressourcen oder andere Dinge.“

Metadaten können dabei, unabhängig von dem Inhalt der Daten, die sie beschreiben, in drei Kategorien unterschieden werden.

- identifizierende Metadaten, wie z.B.: Titel oder Interpret.

- administrative Metadaten, wie z.B.: der Speicherort

- technische Metadaten, wie z.B.: die Abtastrate oder Auflösung

Auch gibt es Metadaten die vom Inhalt der Daten, die sie beschreiben, abhängig sind. Solche Metadaten können zum Beispiel den Inhalt, oder einzelne Teile davon, anhand von Schlagwörtern beschreiben. Ebenfalls können sie etwaige Zusatzinformationen zu Teilen

des Inhalts anhand von Kommentaren oder Hinweisen geben.

Um Mediadateien, wie zum Beispiel MP3-Dateien, mit solchen Metadaten auszustatten, wurde 1996 eine Methode entwickelt die diverse Zusatzinformationen an das Ende der Mediadatei hinzufügt. Dieser sogenannte ID3-Tag kann von den meisten Media Playern ausgelesen und angezeigt werden.

Bei dem Streaming von Audio und Video über das Internet, beinhalten die Metadaten Informationen und Beschreibungen über den Inhalt eines Streams. Mögliche Parameter solcher Metadaten sind zum Beispiel der Künstler/Interpret, der Songtitel, das Album, das Erscheinungsjahr oder das Genre des Inhaltes innerhalb eines Streams. Jedoch können diese Metadaten auch Informationen über den Stream selber, also zum Beispiel den Namen des Internetradios, den Namen des Streams oder aktuelle Nachrichten enthalten.

Im einfachsten Fall sind solche Metadaten, bei der Übertragung von Streams an eine Webseite, in externen XML-Dateien ausgelagert und werden mit Skriptsprachen wie zum Beispiel PHP ausgelesen und auf der Webseite angezeigt.

Aktuelle Streaming-Server sind in der Lage Metadaten innerhalb der Streams die sie versenden mit zu schicken. Dabei werden die Metadaten in einem bestimmtem Intervall in den Stream geschrieben und mit zu dem Benutzer übertragen. Das Intervall, indem die Metadaten in den Stream geschrieben und gesendet werden, ist bei den meisten Streaming-Servern einstellbar. Diese Art der Metadatenübertragung gewährleistet, dass die Metadaten immer (innerhalb des Intervalls) die korrekte Beschreibung zu dem aktuellen Inhalt des Streams aufweisen. Ist zum Beispiel ein Titel zu Ende, werden sofort die Metadaten mit der Information zu dem neuen Titel überschrieben. Auf diese Weise können Media Player und andere Endgeräte die Metadaten aus dem Stream nehmen und in nahezu Echtzeit anzeigen.

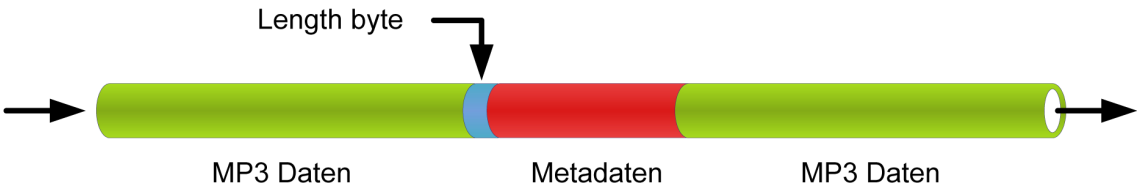


Abbildung 8: Metadaten innerhalb eines Shoutcast Streams

3 Umfrage an moderne Radiomacher

Um im Vorfeld einige Informationen und Anreize für eine geeignete Audio-Streaminglösung zu bekommen, wurde eine Umfrage mit insgesamt 25 Fragen an 60 ausgewählte Teilnehmer geschickt. Diese Umfrage wurde mittels einer Umfragensoftware, mit dem Namen Limesurvey, als Onlineumfrage über eine Webseite in das Internet gestellt.

Um die ausgewählten Teilnehmer, die per E-Mail über diese Umfrage informiert wurden, zu motivieren daran teilzunehmen, beinhaltete dies Umfrage ebenfalls ein kleines Gewinnspiel.

3.1 Erklärung und Aufbau der Umfragenstruktur

Um einen Eindruck über die verwendeten Techniken, auch von nicht deutschen Webradiosendern, zu bekommen, wurde diese Umfrage nicht nur an deutsche Webradio- und Radiosender geschickt, sondern ebenfalls an einige ausgewählte Amerikanische Radiosender. Aus diesem Grunde war es von Nöten die Umfrage zweisprachig zu gestalten. In der Abbildung links ist die Startseite mit der Sprachauswahl Deutsch / Englisch zu erkennen. Die einzelnen Seiten der Umfrage sind im Anhang noch einmal dargestellt.

Warum die Teilnehmer ausgewählt wurden an dieser Umfrage teilzunehmen, stand in der E-Mail die sie erhalten haben. Auf der ersten Seite in der Umfrage stellte sich der Autor noch einmal vor und erklärte warum diese Umfrage überhaupt entstanden ist und welchen Zweck sie erfüllen soll. Auf dieser Seite haben die Teilnehmer die Möglichkeit mit der vollen Umfrage zu starten, nur einen Auszug der wichtigsten Fragen zu beantworten oder die Umfragenseite komplett zu verlassen.

Um die ausgewählten Teilnehmer etwas zu motivieren an der Umfrage teilzunehmen, wurde die dritte Seite der Umfrage mit einem kleinen Gewinnspiel versehen. Dabei erhalten zwei zufällig ausgewählte Teilnehmer, wobei ein amerikanischer und ein deutscher ausgewählt wurde, eine Kaffeetasse mit einem kleinen Dankeschön und dem Logo seines Radiosenders darauf.

Die 25 Fragen innerhalb der Umfrage wurden der Übersicht halber in 4 verschiedene Kategorien - „Der Umgang mit Ihren Angaben und Daten“, Ein paar Allgemeine Fragen über Ihr Webradio“, „Ein paar Fragen über Ihr Streamingangebot“ und „Ein paar Fragen über die Technik die Sie verwenden“ aufgeteilt. Während der Teilnehmer die Umfrage Frage für Frage ausfüllt, wird jeweils nur eine Kategorie mit allen Fragen, die diese Kategorie betreffen, angezeigt.

3.2 Auswertung der Fragen

Die Umfrage beinhaltete insgesamt 25 Fragen zum allgemeinen Aufbau, Fragen über das Streamingangebot und Fragen über die verwendete Technik der Radiosender. In dieser Auswertung sollen jedoch nur Fragen bezüglich der verwendeten Technik und einige Fragen über das Streamingangebot einfließen, da eine Auswertung der kompletten Daten diese Arbeit sprengen würde.

3.2.1 Verwendete Streaming-Server

In Anbetracht der Tatsache, dass die befragten Teilnehmer ein Mix aus klassischen Sendern, die primär über terrestrischen Funk senden, und reinen Webradio Sendern waren, kristallisiert sich bei der Frage welche Streaming-Server sie verwenden ein eindeutiger Trend heraus.

Da der SHOUTcast Server nicht in der Lage ist einen Live-Stream von zum Beispiel einer Soundkarte zu erzeugen, sondern lediglich dazu verwendet werden kann lokal abgelegte Musikdateien anhand von Playlisten zur Verfügung zu stellen, ist aus oben dargestellten Diagramm ersichtlich, dass die reinen Webradiosender ausschließlich den SHOUTcast-

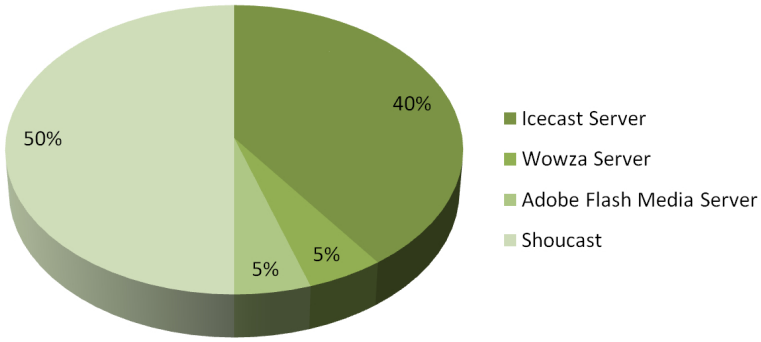


Abbildung 9: Welche Streaming-Server verwenden Sie?

Server verwenden um ihr Programm an die Benutzer zu übertragen. Sender, die ihr für den terrestrischen Funk produziertes Programm im Internet als Stream zur Verfügung stellen, verwenden ausschließlich den Icecast Server und ergänzen ihr Angebot mit dem Wowza Media oder dem Adobe Flash Media Server.

3.2.2 Verwendete Betriebssysteme

Bei der Frage nach den Betriebssystemen die für die Streaming-Server verwendet werden, spalten sich die Antworten nicht so deutlich voneinander ab. Lediglich 30% der befragten Parteien nutzen das Betriebssystem Microsoft Windows, obwohl dieses Betriebssystem gerade für den Bereich der Erstellung eines reinen Webradios durch die Vielzahl an zu verwendeter freier und kostenloser Software und nicht zuletzt der Bedienerfreundlichkeit, geeignet ist.

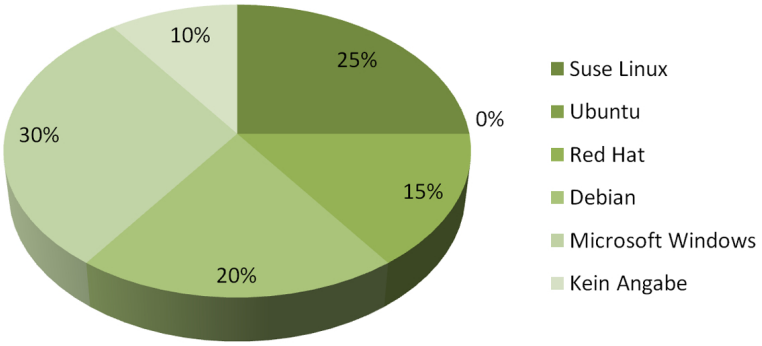


Abbildung 10: Diagramm für „Welches Betriebssystem verwenden Sie?“

Die Mehrzahl der Befragten nutzen Linux und Linux-Derivate um ihr Programm zu wandeln und um es per Stream in das Internet auszustrahlen. Der Vorteil dieser Betriebssysteme beruht hier auf der Tatsache, dass für diese Plattformen sehr leistungsstarke Live-Encoder und Streaming-Server zur Verfügung stehen. Diese Vorteile sind gerade bei dem Umwandeln eines schon existierenden Signals von Nutzen, da so eine fast autarkes und automatisches System geschaffen werden kann, welches nach Inbetriebnahme wenig Pflege bedarf.

3.2.3 Verwendete Encodersysteme

Obwohl mit dem Einsatz von Hardware-Encodern ein sehr stabiler und ausfallsicherer Workaround geschaffen werden kann, benutzen lediglich 10% der Befragten eine solche Technik. Die Mehrzahl der befragten Teilnehmer gab an, dass sie Encoder wie den Shoutcast (30%) und ähnlich angelehnte Encoder wie den Sam Broadcaster (15%) benutzen. Die Stärken dieser Encoder liegen eindeutig auf der Bedienerfreundlichkeit der Software, da sie bequem von jedem PC zum Beispiel mittels eines Winamp Players mit den zu encodierten Daten versorgt werden können.

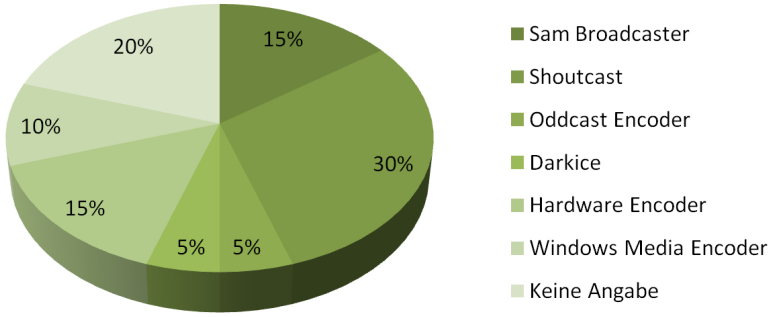


Abbildung 11: Diagramm für „Welche Encoder verwenden Sie?“

3.2.4 Bearbeitung der Streams

Um bei der Übertragung von Musik über das Internet ein Qualitativ hochwertiges Signal zu erhalten, besteht die Möglichkeit das ausgehende Signal vor dem Versenden noch einmal zu bearbeiten. Dies ist unter anderen von Nöten, wenn das Signal direkt von der Sendeschiene eines klassischen Radios abgegriffen wird.

Umso mehr überrascht es, dass nur 50% der Befragten ihr Signal vor der Ausstrahlung

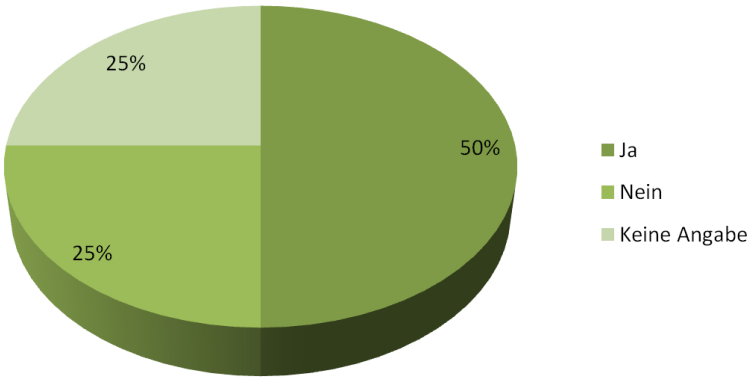


Abbildung 12: Diagramm für „Bearbeiten Sie vor dem Versenden Ihr Signal?“

noch einmal anpassen. Dabei gaben alle die diese Frage mit Ja beantworteten an, dass sie ihr Signal in der Lautstärke ändern. 50% gaben ebenfalls an, dass sie ihr Signal durch die Verwendung eines digitalen Signalprozessors noch einmal verfeinern.

3.2.5 Einfügen von Metadaten

Eine gute Möglichkeit um einen Mehrwert des vorhandenen Streamingangebotes zu schaffen, ist die Änderung der Metadaten innerhalb eines Audio-Streams. So verwundert es nicht das 65% der Befragten die Metadaten innerhalb ihrer Streams bearbeiten.

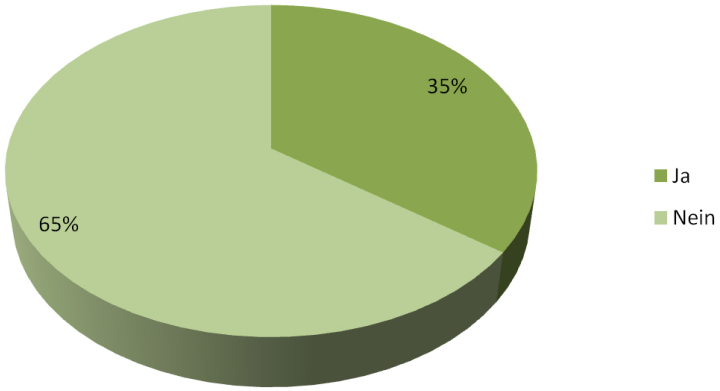


Abbildung 13: Diagramm für „Ändern Sie die Metadaten Ihrer Streams?“

Dabei gaben alle die diese Frage mit Ja beantworteten an, dass sie Informationen zur laufenden Sendung und den Namen des Senders mit in ihre Streams einfügen.

4 Vorüberlegungen

Um ein Konzept für eine Audio- und Video Streaminglösung überhaupt erstellen zu können, ist es nicht nur wichtig die notwendigen Grundlagen und Begriffe zu kennen, sondern es ist ebenfalls von Nöten im Vorfeld zu klären welche, Eigenschaften und Fähigkeiten eine solche Lösung beinhalten soll. Das folgende Kapitel zeigt einige Vorüberlegungen zu einer solchen Lösung auf.

Um den Benutzern einen optimalen Zugang und eine optimale Erreichbarkeit zu einem Streamingangebot bieten zu können, müssen im Vorfeld einigen Überlegungen zur Produktion, der Bereitstellung und Verbreitung der Inhalte getroffen werden. Wichtige Überlegungen dabei sind es, festzulegen über welche Techniken der Benutzer Zugriff auf die medialen Inhalte erhält, wie hoch die Ausfallsicherheit der Streamingsysteme sein soll und wie viel Mehrwert dem Benutzer über das Streamingangebot hinaus bei der Nutzung des Angebotes zur Verfügung steht.

Um den Benutzern einen von ihrer zugrundeliegenden Rechner- und Betriebssystemarchitektur unabhängigen Zugang zu den Inhalten bieten zu können, benötigt eine optimale Lösung ein Angebot welches sich leicht über standardisierte Übertragungswege transportieren lässt. Ein optimales Angebot ist in der Lage sich in eine Website integrieren zu lassen und zusätzlich für die meisten Media Player verständlich zu sein. Da die meisten Betriebssysteme eine Browsersoftware bieten, könnte der Benutzer die Inhalte über den Besuch der Website, über einen dort integrierten Player, nutzen. Benutzt das Angebot ein Streaming Format welches von verschiedenen Media Playern abgespielt werden kann, unterliegt der Benutzer zusätzliche keinem Zwang, was die benötigte und zu benutzende externe Abspielsoftware betrifft.

4.1 Vorüberlegungen für die Audio-Streams

Eine Methodik im Bereich des Audio-Streaming, die sich für die Verbreitung über Media Player und innerhalb von Webseiten gut eignet und auch bewährt hat, ist die Verbreitung der Inhalte im MP3-Format. Die meisten Media Player können dieses Format ohne Einschränkungen abspielen. Ein MP3-Stream hat beispielsweise im Gegensatz zu einem Windows Media Stream, den Vorteil, dass er plattformübergreifend funktioniert und so auch unter Verwendung verschiedenster Media Player Anwendung findet.

Die Hauptaufgabe einer Audio Streaminglösung für ein intermediales Redaktionssystem sollte aber sein, dass die Inhalte dem Benutzer direkt auf der Portalseite zur Verfügung stehen. So sollte der Benutzer die Möglichkeit haben während seines Aufenthaltes auf dem Portal das Webradio nutzen zu können.

Basierend auf einem MP3-Stream wäre die Integration eines Flash Players für die Audio-

Streams in eine Webseite für eine optimale Streaminglösung von Nutzen. Die Integration eines Flash Players wäre für die Benutzer von großem Vorteil, da sie um diese Inhalte benutzen zu können ebenfalls an keine Plattform gebunden wären. Nahezu jedes Betriebssystem bietet heutzutage Browsersoftware die in der Lage ist Flash zu betrachten und bietet somit automatisch Zugang zu den medialen Inhalten auf der Webseite.

Ebenso ist ein Flash Player in seiner Gestaltung und im seinem Funktionsumfang relativ einfach anpassbar. So könnte ein Flash Player individuell auf die Bedürfnisse des Portals, und sei es nur sein Aussehen welches an das Corporate Design angelehnt ist, modifiziert und optimiert werden.

Die Entwicklung und Anwendung einer Audio-Flash-Streaminglösung für das Portal würde auch bei der Bereitstellung von Videoinhalten von großem Nutzen sein. Ein Flash Player ist ebenfalls in der Lage On-Demand und Live-Videos abzuspielen. Neben den schon genannten Vorteilen einer Streaminglösung mittels Flash, hätte der Benutzer so ebenfalls die Möglichkeit auf der Portalseite die Videoangebote mittels eine Flash Players zu nutzen.

Eine weitere notwendige Überlegung ist es zu definieren in welchen Qualitäten dem Benutzer die Inhalte zur Verfügung stehen sollen. Basierend auf der Anbindung an das Internet, haben die Benutzer die Möglichkeit qualitativ hochwertige Streams empfangen zu können oder nicht. Auf der einen Seite ist es also von sehr großem Vorteil sein Angebot möglichst qualitativ hochwertig bereitzustellen, auf der anderen Seite besitzt eben nicht jeder Benutzer eine starke und stabile Breitbandverbindung und kann somit keine qualitativ sehr hochwertige Streams empfangen.

Eine solche Überlegung ist besonders für das Streaming von Videoinhalten wichtig, da dort wesentlich mehr Daten an den Benutzer gesendet werden müssen. So kann es von großem Vorteil sein die selben Angebote in verschiedenen Qualitäten zur Verfügung zu stellen. Man sollte also nach einer Streaming-Lösung streben, die dem Benutzer ermöglicht die Inhalte nicht nur in bester und höchster Qualität nutzen zu können, sondern ebenfalls ein Angebot bereitstellen, welches jeder Benutzer nutzen kann, insofern keine Breitbandverbindung besteht.

Neben dem reinen Streamingangebot, kann es auch von Vorteil sein diverse Zusatzinhalte anzubieten. Zum Beispiel ist es bei dem Senden von Musik von großem Vorteil zusätzlich noch Informationen über die gerade wiedergegebenen Inhalte zur Verfügung zu stellen. So könnte dem Benutzer ein Mehrwert geboten werden, indem er sieht welche Musik gerade gespielt wird, welcher Künstler und welcher Song gerade läuft. Aber auch Informationen wie aktuelle Nachrichten, Spielstände von Sportereignissen oder ein Kauf-Angebot für den Erwerb einer CD, von dem Künstler der gerade zu hören ist, sind denkbar.

Die Bereitstellung solcher Metadaten an einen Flash Player ist auf mehreren Wegen realisierbar. Im einfachsten Fall liest der Webserver, auf dem das Portal des intermedialen Redaktionsystem hinterlegt ist, eine XML-Datei aus und stellt so die Informationen im Portal dar. Jedoch lässt dieser Lösungsansatz keine Echtzeitsteuerung der Metadaten zu. So kann es bei dieser Lösung vorkommen, dass eine Information im Portal angezeigt wird die schon seit einiger Zeit veraltet ist. Problematisch ist auch ein Ausfall dieses Arbeitsschrittes, da erst relativ spät bemerkt wird, dass es zu einem Fehler oder einem Ausfall durch zum Beispiel eine Überlastung des Web-Servers gekommen ist.

Ein anderer möglicher Weg ist es, die Metadaten innerhalb des Streams mit zu senden. Dadurch ist gewährleistet, dass die zu einem Titel verfügbaren Informationen nur dann angezeigt werden, wenn der Titel auch gerade verwendet wird. Um diese Möglichkeit benutzen zu können, wird jedoch ein Flash Streaming Server benötigt, der in der Lage ist die Metadaten auf diesem Wege an den Flash Player zu schicken.

4.2 Vorüberlegungen für die Video-Streams

Da dem Benutzer auf dem resultierendem Portal einzelne Video-Clips zur Verfügung stehen sollen und zusätzlich noch die Möglichkeit geboten werden soll das Live-Programm von dem TV-Sender MW*DIGITAL* als Live-Stream zu nutzen, ist es von Vorteil an dieser Stelle die Vorüberlegungen für das Video-Streaming in zwei Bereiche aufzuteilen. Dem Bereitstellen der Video-Clips und dem Bereitstellen des Live-TV-Signals.

Das Bereitstellen von einzelnen Video-Clips auf dem Portal entspricht einer On-Demand-Streaminglösung oder einem Progressivem Download, da die einzelnen Clips auf dem Server als Dateien vorliegen und je nach Bedarf von einem Benutzer abgerufen werden können. Das Streamen eines Livebildes von einem TV-Sender ist dagegen nur mit einer Echtzeit-Streaminglösung zu bewältigen, da hier die Daten von einem Encoder in Echtzeit umgewandelt und über einen Streaming-Server zur Verfügung gestellt werden müssen.

4.2.1 Bereitstellen der Video-Clips

Auf dem resultierendem Portal sollen dem Benutzer einzelne Video-Clips zur Verfügung stehen, die er bei Bedarf starten, stoppen, pausieren und spulen kann. Ein solcher Funktionsumfang kann über verschiedene Ansätze erreicht werden. Zum Beispiel ist es möglich mit diversen Browser-Plugins, die einen integrierten Player auf der Webseite erzeugen können, ein Windows Media Video (WMV) oder ein QuickTime Movie (MOV) abzuspielen. Diese Methoden haben jedoch den Nachteil, dass die benötigten Browser-Plugins extra für das Betrachten der Videos installiert werden müssen und die sich auf diese Weise in die Webseite integrierten Player sich wenig oder gar nicht in Form, Aussehen und Funktionsumfang ändern lassen.

Eine weitere Möglichkeit ist das Bereitstellen von Videos innerhalb eines Flash Players. Dieser Lösungsansatz hätte Vorteile in der Flexibilität des resultierenden Players, denn bei einem Video-Flash Player lassen sich, ähnlich wie bei einem Flash-Audio Player, Aussehen und Funktionsumfang im Gegensatz zu oben genannten Playern anpassen. Zusätzlich würden bei diesem Ansatz Vorteile in der großen Verbreitung des benötigten Browser Plugins entstehen. Das benötigte Plugin, der Adobe Flash Player, ist laut Adobe selber auf 98% der Rechnersysteme installiert.

Bei der Bereitstellung der Inhalte über eine Video-Flash-Streaminglösung, können zwei Workarounds in Betracht gezogen werden. Ein Video-Flash Player ist in der Lage Videos, die in dem von Adobe entwickelten Flash Videoformat (.flv) auf dem Server hinterlegt sind, direkt abzuspielen. Dabei stellt diese Methode die Video-Clips über einen progressiven Download zur Verfügung. Ein weiterer möglicher Workaround ist die Verwendung eines Flash Media Streaming-Servers. Dabei werden auf dem Server hinterlegte Videodateien über den Streaming-Server in das benötigte Flash Videoformat umgewandelt und an den Benutzer übertragen.

Ein großer Vorteil, der bei der Nutzung einer Streaminglösung mittels eines Flash Media Streaming-Servers entsteht, ist die Fähigkeit, dass in dem bereitgestellten Video-Clip in nahezu Echtzeit hin und her gesprungen werden kann. Diese Fähigkeit ist bei der Lösung ohne einen Streaming-Server erst nach dem kompletten Download des Inhaltes gewährleistet.

Bereitstellung des Content

Um die Video-Clips über das Portal den Benutzern zur Verfügung stellen zu können, müssen diese im Voraus in ein für die Wiedergabe benötigtes Format umgewandelt und auf dem Server hinterlegt werden. Dieser Arbeitsschritt findet normalerweise an dem lokalen Computer eines Erstellers statt, indem er die Inhalte mit einer geeigneten Software in das entsprechende Format umwandelt und im Anschluss durch das Hochladen auf den Server bereitstellt.

In Anbetracht der Tatsache, dass bei der Pflege und Aktualisierung eines intermedialen Redaktionssystemes verschiedene Redakteure aus verschiedenen Bereichen mit unterschiedlichem Kenntnisständen tätig sein werden, wäre es von Vorteil wenn dieser Arbeitsschritt für die Redakteure etwas vereinfacht werden würde. So könnten die Redakteure das Videomaterial auf den Server hochladen und der Streaming-Server selber hätte die Aufgabe die notwendigen Umwandlungen vorzunehmen.

Für einen solchen Workaround kommen verschiedene Transkodierungsprogramme, die serverseitig ausgeführt werden können, in Frage. Einige kostenfreie Open-Source Transkodie-

rungprogramme sind zum Beispiel FFmpeg, Mencoder und Transcode, wobei Mencoder und Transcode auf FFmpeg aufbauen.

Die Sothink Video Encoder Engine ist eine proprietäre Software um Videos serverseitig zu wandeln, dabei ist sie in der Lage auf Windows- als auch Linux-basierenden Servern eingesetzt zu werden. Eine weitere proprietäre Software, die in der Lage ist Videos in das Flash Format zu konvertieren, ist On2s Flix. On2s Flix ist ebenfalls für Windows und Linux erhältlich und ermöglicht, wie die Sothink Video Encoder Engine, die Anbindung je nach Plattform durch formale Sprachen wie ASP, ASP.net und PHP.

Ein solcher Lösungsansatz könnte neben der vereinfachten Bedienung durch die Redakteure den Vorteil haben, dass das originale Videomaterial, welches meist eine bessere Qualität aufweist, nicht verloren geht und zusätzlich noch als Download auf dem Portal angeboten werden kann. Jedoch ist dabei zu bedenken das die Speicherung zusätzlich Platz auf dem Server benötigen würde und die Zeiten für das Hochladen von Videos in besserer Qualität meist sehr viel höher sind als das Hochladen eines schon umgewandeltem Videos.

4.2.2 Live-Streaming des TV-Signals

Neben dem Streamingangebot der Video-Clips auf dem Portal, soll zusätzlich noch das Live TV-Signal des Senders MW*DIGITAL* mit in das Streamingangebot einfließen. Um eine solche Funktionalität zu bewerkstelligen, bedarf es ähnlich dem Live-Audio-Streaming einen Workaround mit Live-Encodern und erfordert den Einsatz von Streaming-Servern.

Die Thematik Live-Video-Streaming steht noch relativ am Anfang ihrer Entwicklung. Um einen solchen Workaround zu bewerkstelligen, gibt es die Möglichkeit den kompletten Ablauf mit spezieller und teurer Hardware zu gestalten oder einen Kompromiss aus geeigneter Hardware und geeigneter Software zu finden.

Für die Entwicklung der Video-Streaminglösung des intermedialen Redaktionssystems, wäre wohl ein Prototyp aus geeigneter PC-Hardware und entsprechender Software der beste Weg um das Livesignal zu den Benutzern zu übertragen.

5 Anforderungen

Wurden alle notwendigen Vorüberlegungen für eine optimale Lösung des Streamingangebotes getroffen, ist es notwendig aus diesen Überlegungen Anforderungen zu formen. Das folgende Kapitel beschreibt die Anforderungen an die Audio- und Video-Streaminglösung die innerhalb des Konzeptes eingehalten werden sollten.

5.1 Anforderungen an eine Streaminglösung für Audio

Bei einem intermedialen Redaktionssytem liegt die Hauptaufgabe einer Streaminglösung bei der Bereitstellung eines Streamingangebotes direkt auf einer Webseite. Egal ob Audio oder Video, beides sollte sich auf dem resultierendem Portal einheitlich präsentieren lassen und dabei nicht auf verschiedene rudimentäre Techniken, und damit verschiedene zu benutzende Abspielsoftware, zurückgreifen.

So wäre es eine Anforderung an eine optimale Streaminglösung, wenn dem Benutzer eine einheitliche Plattform auf dem resultierendem Portal zur Verfügung stehen würde und er nicht erst verschiedene Plugins oder Player für das Hören von Audio und das Betrachten von Video benötigt. Ein Ziel für ein intermediales Redaktionssystem soll also sein, eine einheitliche Plattform für die Präsentation der Audio- und Videodaten zu finden.

5.1.1 Die Audio-Streams

Heutzutage reicht die Bereitstellung eines einzelnen Audio-Streams über einen einzelnen Übertragungsweg meist nicht mehr aus. Die meisten Benutzer wünschen und fordern die Möglichkeit solche Inhalte an verschiedenen Orten und mit verschiedenen Endgeräten nutzen zu können. So sollte man bei der Überlegung welche Techniken man für die Bereitstellung der Streams für die Benutzer verwendet, auf mehrere Techniken zurückgreifen und ein Angebot erstellen welches mit verschiedenen Endgeräten funktioniert.

Eine Anforderung an ein intermediale Redaktionssytem ist somit, dass die Inhalte über mehrere Übertragungswege, und somit auch für mehrere Endgeräte, zur Verfügung gestellt werden können. Ideal wäre nicht nur die Realisierung eines Übertragungsweges direkt an einen in einer Webseite integrierten Player, sondern die zusätzliche Realisierung einer Übertragung an die gängigsten Media Player wie Winamp oder den Windows Media Player. Mit diesen zwei sich ergänzenden Techniken wird der Grundbedarf eines Streamingangebotes bereits sehr gut abgedeckt. Die Benutzer sind in der Wahl eines für sie geeigneten Media Players frei, können aber auch ohne Media Player auf die Inhalte über die Webseite zugreifen.

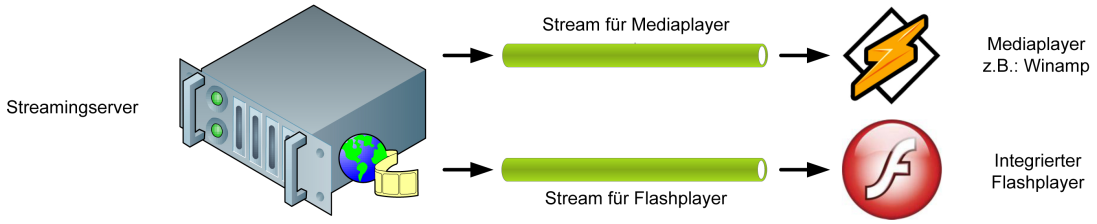


Abbildung 14: Streams an mehrere Endgeräte senden

Um den Benutzern auch einen Mehrwert über die reinen Audio-Streamingangebote hinaus bieten zu können, empfiehlt es sich über die Metadaten der Audio-Streams Zusatzinformationen zu senden. So wäre ein Benutzer in der Lage anhand der Metadaten eines Streams sofort zu sehen welcher Titel oder welche Sendung gerade läuft. Eine Versendung solcher zusätzlichen Informationen über die Metadaten bietet sich an, da ein Flash Player diese Informationen aus dem Stream entnehmen und anhand eines Tickers anzeigen können. Ebenfalls lesen die meisten Media Player diese Metadaten aus und stellen die Informationen im Player grafisch dar.

Somit ist eine weitere Anforderung an die Streaminglösung, die Audio-Streams des intermedialen Redaktionssystems mit Metadaten auszustatten, die Informationen zu dem gerade laufendem Song beinhalten. Diese Informationen sollten als Ticker im Player auf der Webseite dargestellt werden und halten somit auch automatisch Einzug in die meisten Media Player.

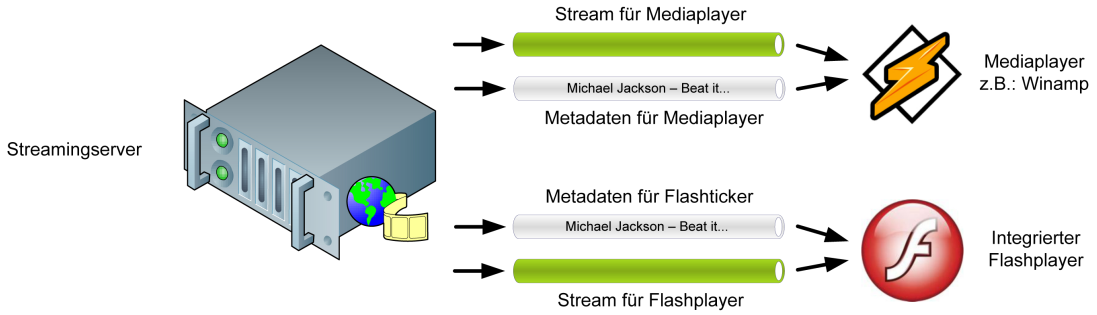


Abbildung 15: Streams mit Metadaten an mehrere Endgeräte senden

5.1.2 Die Metadaten

Die Metadaten in den einzelnen Streams sollten einen Mehrwert für den Benutzer bieten. Im einfachsten Fall steht hier also nicht nur der Name des Streams der gerade von Benutzer genutzt wird, sondern es befinden sich aktuelle Informationen über den Interpreten

und den Titel, der gerade läuft, innerhalb dieser Metadaten. Denkbar wäre auch eine Integration von Information welche Sendung und welche Moderatoren gerade zu hören sind. Diese Informationen sollten nicht nur für die Media Player zur Verfügung stehen, sondern auch in dem Flash Player auf der Webseite angezeigt werden.

Eine Anforderung an eine optimale Streaminglösung ist es also, ein System zu entwickeln um die Metadaten der Streams automatisch an den aktuellen Inhalt der Streams anzupassen. Ebenfalls interessant wäre ein zusätzliches Modul welches den Redakteuren und Moderatoren erlaubt eigene Informationen in die Metadaten zu integrieren.

5.1.3 Der Audio Flash Player für das Portal

Um den Benutzern einen optimalen Zugang zu den Inhalten des Audio-Streaming Angebot auf dem resultierendem Portal bieten zu können, empfiehlt es sich einen Flash Player in den oberen Bereich der Webseite zu integrieren. Dieser Player sollte auf jeder Unterseite des Portales angezeigt werden und alle Funktionen beinhalten die das Audio-Streamingangebot zur Verfügung stellt. Der Benutzer sollte hier nicht nur den Flash-Stream benutzen können, sondern auch die Streamingangebote für die Media Player, wie zum Beispiel den Winamp und Windows Media Player, per Knopfdruck erreichen können. So wäre ein Benutzer in der Lage bei seinem Besuch, während er über die Text und Bild Inhalte stöbert, die Audio-Streams als Begleitmedium über den integrierten Flash Player zu nutzen. Oder er nutzt nur das reine Audioangebot, über den Start eines externen Media Player.

Der Flash Player für die Audio-Streams sollte sich nahtlos in das Layout des Portales einfügen und dem Corporate Design des resultierendem Portals entsprechen. So bietet er mit seinem Funktionsumfang und seinem optischem Auftreten ein Highlight in jedem einzelnem Bereich des Portales. Ebenfalls sollten in dem Flash Player die aktuellen Titelinformationen angezeigt werden. Denkbar ist eine Art Ticker indem der aktuell zu hörende Künstler und der Name des Songs als Laufschrift läuft.

Eine Anforderung an die Streaminglösung ist somit, einen Audio Flash Player zu entwickeln der sich permanent im oberen rechten Bereich der Webseite befindet, einen Lauftext-Ticker für die Anzeige der aktuellen Metadaten besitzt und eine Symbolleiste bietet über die die Streams für andere Media Player (z.B.: Winamp, Windows Media Player) geöffnet werden können.

In der bevorstehenden Grafik ist eine beispielhafte Umsetzung eines Flash Players zu sehen. Dieser Entwurf bezieht sich vom Design und den Farben her auf das Logo von 99drei

Abbildung 16: Beispielhafte Umsetzung eines Flash-Audioplayers

- Radio Mittweida. Zu erkennen sind rechts unten drei Buttons (Winamp, Windows Media Player und iTunes). bei einem Mausklick auf die Buttons wird versucht der jeweilige Media Player zu starten und es wird versucht den Stream automatisch wiederzugeben. Weiterhin befinden sich drei Steuerelemente (Play, Stop und Pause) im linken unterem Bereich des Players. Mit diesen Elementen wird der Flash-Stream gestartet, gestoppt oder pausiert. Zusätzlich befindet sich im oberen Teil noch eine Lauftext-Anzeige, in der im einfachsten Falle, der aktuelle Titel als Lauftext angezeigt werden kann.

5.1.4 Streaming-Server für Audio

Der Streaming-Server für die Audio-Streams sollte, neben der Fähigkeit mehrere Streams in verschiedenen Qualitätsstufen zu versenden, auch in der Lage sein diverse Metadaten in die Streams zu schreiben und mit dem Stream gemeinsam zum Benutzer zu übertragen. Da das MP3-Format für Streams sehr gebräuchlich ist und nahezu jeder Media Player diesen Stream wiedergeben kann, empfiehlt es sich die Streams im MP3-Format zu produzieren und zu versenden.

Eine Anforderung ist es, eine Streaming-Server Software zu finden die mehrere MP3-Streams produzieren kann und in der Lage ist die Metadaten in den Streams zu verändern.

Um das Streamingangebot auch mittels eines integrierten Flash Players auf dem resultierendem Portal zur Verfügung stellen zu können, wird zusätzlich noch eine Streaming-Server Software benötigt die die vorhanden Streams aufbereitet, umwandelt und diese als Flash-Streams an den integrierten Flash Player sendet.

Eine weitere Anforderung ist, mittels geeigneter Streaming-Server Software einen Lösungsweg und einen Workaround zu finden, um die Streams für die Media Player und die Streams für den Flash Player zu produzieren, mit Metadaten zu versehen und an die entsprechen-

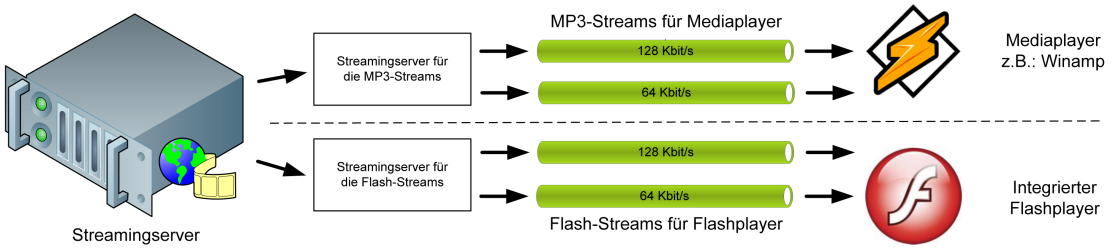


Abbildung 17: MP3-Stream und Flash-Stream an mehrere Endgeräte senden

den Endgeräte zu senden.

Ein Streaming-Server sollte in der Lage sein, auf einen Ausfall oder eine Überlastung eines Streams zu reagieren. Er sollte Mechanismen zur Verfügung stellen, um zum Beispiel bei einem Fehlerfall den Ausfall zu melden und eine entsprechende Notlösung bereitstellen. So könnte ein Streaming-Server mit einer Fall-Back Technologie die Überlastung eines Streams erkennen und Benutzer auf einen anderen Stream (z.B.: einen Stream mit einer geringeren Qualität) umleiten. Bei einem totalen Ausfall der Live-Streams, wenn zum Beispiel keine Audiodaten mehr empfangen, encodiert oder verschickt werden, könnte der Server automatisch jeden Benutzer auf einen On-Demand-Stream leiten, auf dem eine Art Not-Schleife läuft.

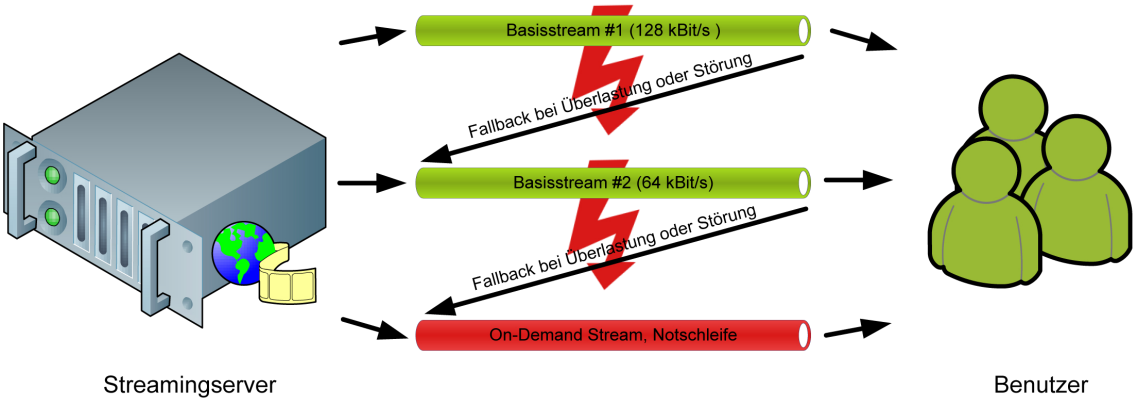


Abbildung 18: Schematische Darstellung eines Fall-Back auf einen Not-Stream

Somit muss eine Streaming-Software gefunden werden, die notwendige Mechanismen zur Verfügung stellt um bei einem Ausfall den Fehler schnellstmöglich zu bemerken und somit den Schaden so gering wie möglich zu halten.

5.2 Anforderungen an eine Streaminglösung für Video

Resultierend aus den Vorüberlegungen zu einer geeigneten Video-Streaminglösung bilden sich für das Videoangebot auf dem Portal zwei Bereiche, die unterschiedlichen Anforderungen unterliegen, heraus. Dabei müssen die Anforderungen für das On-Demand Angebot und die Anforderungen des Live-Streamingangebotes voneinander unterschieden werden.

5.2.1 Bereitstellen der Videoclips

Um die Video Clips auf dem Portal bestmöglich präsentieren zu können, wird eine einheitliche Plattform benötigt, die nahezu jedem Benutzer von Anfang an zur Verfügung steht. Um den Benutzern eine einfache Möglichkeit bieten zu können die Inhalte zu nutzen, empfiehlt es sich auf weit verbreitete Techniken zurückzugreifen und auf Techniken zu verzichten die dem Benutzer ein gewisses Maß an Computerkenntnissen abverlangen. So wäre es von großem Vorteil auf die notwendige Installation eines Extra Plugins durch den Benutzer zu verzichten und eher auf Techniken zurückzugreifen die der Benutzer im Zuge von anderen Anwendungen bereits besitzt. Eine Anforderung an die Video-Streaminglösung besteht darin die Videoclips über eine weitverbreitete und einfach zu nutzende Technologie zur Verfügung zu stellen.

Die Video Clips, die später auf dem Portal präsentiert werden sollen, sind in ihrer Dateigröße und Länge relativ groß und in der Qualität sehr hochwertig. Um den Benutzern unnötige Wartezeiten während eines progressiven Downloads zu ersparen und die Möglichkeit des sofortigen Spulens sowie des Hin- und Herspringens bieten zu können, empfiehlt sich an dieser Stelle der Einsatz eines On-Demand Flash Streams über einen Flash Media Server. Eine Anforderung an das System ist, dass die Video Clips nicht über einen progressiven Download zu den Benutzer übertragen werden, sondern unter Zuhilfenahme eines Streaming-Servers gesendet werden.

Ebenfalls ist es für das Betreiben einer Video-Streaminglösung auf der Portalseite von Nöten, einen geeigneten Weg zu finden um die Inhalte durch den Einsatz von Redakteuren zur Verfügung stellen zu können. Hierbei sollten Techniken und Mechanismen zur Verfügung stehen, die es auch unerfahrenen Redakteuren ermöglichen die Inhalte zu veröffentlichen. Eine Anforderung an das Bereitstellen von Videoclips auf dem Portal wäre somit ein geeigneter Workaround, der es auch unerfahrenen Redakteuren ermöglicht Inhalte zu veröffentlichen.

5.2.2 Live-Streaming des TV-Signals

Die Hauptanforderung bei dem Streamen eines Livebildes innerhalb des intermedialen Redaktionssystem ist, einen Workaround zu erstellen der den finanziellen Aufwand und den resultierenden Mehrwert durch ein einziges TV-Signals abwiegt. Den Workaround allein durch den Einsatz von professioneller Hardware zu bewältigen, wäre ein einfacher aber

auch sehr kostenintensiver Weg.

Wie bei der Bereitstellung der Video Clips, sollten hier ebenfalls Anforderungen an die Bereitstellung bei dem Benutzer aufgestellt werden.

5.3 Zusammenfassung der Anforderungen

Die dargestellten Überlegungen führen somit zu einer Liste der folgenden Anforderungen:

Für das Streamen der Audiodaten ist ein MP3-Stream und Flash-Stream über Streaming-Server in verschiedenen Qualitäten mit einfachen und zusätzlichen Metadaten anzustreben. Aus Kostengründen sollte ein Software-Encoder bevorzugt werden. Gleichzeitig muss der Streaming-Server über eine Ausfallsicherung verfügen.

Der Flash Audio Player ist dem Corporate Design des resultierendem Portal anzupassen und bietet neben der Wiedergabe der Flash-Streams die Möglichkeit die MP3-Streams zu starten.

Video-Streams müssen durch Flash-Streams anhand von On-Demand- und Live-Streaming über einen auf der Webseite integrierten Flash Player dargestellt werden. Dabei werden die Streams über einen Flash Media Server dem Benutzer zur Verfügung gestellt.

6 Konzept

Anhand der getroffenen Vorüberlegungen und den aufgestellten Anforderungen an eine Streaminglösung für ein intermediales Redaktionssystem, können verschiedene Konzepte und Workarounds entwickelt werden. In den nachfolgenden Kapiteln, werden einige mögliche Konzepte vorgestellt.

6.1 Konzept der Streaminglösung für Audio

Eine optimale Lösung um das Audio-Streaming Angebot, für ein aus dem intermediale Redaktionssystem entstehendem Portal, effektiv zu den Benutzern übertragen zu können, ist also ein Mix zwischen Streams in verschiedenen Qualitäten die auf der Webseite innerhalb eines Flash Players eingebunden werden können und Streams in verschiedenen Qualitäten für diverse Media Player wie zum Beispiel den Winamp oder den Windows Media Player.

Aktuell ist das versenden von Streams im MP3-Format sehr weit verbreitet, somit besteht eine gute Chance das Streams die in diesem Format produziert und versendet werden an den meisten Endgeräte benutzt werden können. Daher empfiehlt es sich die Audio Streams des intermedialen Redaktionssystems für die Media Player im MP3-Format zu produzieren und zu veröffentlichen. Um dieses Konzept umsetzen zu können, wird eine Streaming Server Software benötigt die nicht nur das MP3-Format unterstützt, sondern ebenfalls in der Lage ist mehrere Streams in verschiedenen Qualitäten zu erzeugen, zu verarbeiten und im Anschluss zu versenden.

Da es momentan keinen Flash Player gibt, der die Metadaten aus einem reinem MP3-Stream lesen kann, jedoch die Möglichkeit besteht die Metadaten aus einem Flash-Stream auszulesen, empfiehlt es sich den Flash Player des Portals mittels einem Flash Streaming Server mit Flash-Streams zu versorgen. Da für das Video-Streaming ebenfalls ein Flash Streaming-Server benötigt wird, sollte dieser Server ebenfalls die vorhandenen MP3-Audio-Streams wandeln und als Flash-Stream an den Audio Flash Player in der Webseite schicken.

6.1.1 Die Audio-Streaming-Server

Die Anforderungen für die Audio Streams lassen sich leider nicht mit nur einer Streaming-Server Software realisieren. Eine Software welche die Fähigkeit besitzt MP3-Streams und gleichzeitig Flash-Streams zu produzieren, gibt es nicht.

Audio-Streaming-Server für die Basis Streams

Der Icecast Server ist eine Open Source Streaming Media Server Software die verschiedene Formate verbreiten kann. Unter anderem bietet der Server die Möglichkeiten MP3, Ogg oder AAC-Inhalte veröffentlichen zu können und ist in der Lage mit der Hilfe von so genannten Sourceclients Live-Inhalte von einer Soundkarte zu verbreiten. Ebenfalls ist er in der Lage statische Inhalte für die Benutzer zur Verfügung zu stellen und ermöglicht damit die Erstellung eines On-Demand Streamingangebotes. In Zusammenarbeit mit mehreren Sourceclients, ist der Icecast Server in der Lage auch von mehreren Audio-Quellen Daten zu erhalten und diese in verschiedene Streams zu verteilen.

Anhand dieser Eigenschaften bietet sich der Icecast Server sehr gut für die Verbreitung von Liveinhalten über MP3-Streams in verschiedenen Qualitäten oder mit unterschiedlichem Inhalt an.

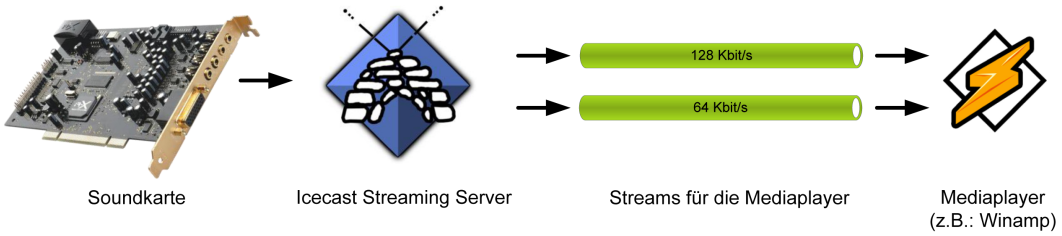


Abbildung 19: Streaming von einer Soundkarte mittels dem Icecast Server

Ebenfalls bietet der Icecast Server Mechanismen um die Metadaten innerhalb eines Stream zu editieren, somit ist es möglich jedem Stream individuelle Metadaten mit auf seinem Weg zum Player zu geben.

Durch seine Fall-Back Technologie können Benutzer, die zu einen Streams verbinden wollen der bereits voll oder gerade durch eine Störung nicht verfügbar ist, auf einen anderen Stream umgeleitet werden. Somit bietet er notwendige Mechanismen die den Streaming-Server vor Überlastung schützen und bei einem auftretenden Fehler einen totalen Ausfall des Streamingangebotes verhindern können.

Anhand seiner Eigenschaften und Fähigkeiten, drängt sich der Icecast Streaming Server direkt an die Spitze der Auswahl, wenn es um eine Entscheidung für einen geeigneten Streaming-Server für die Produktion der Basis Streams im MP3-Format des intermedialen Redaktionssystem geht. Leider bietet der Icecast Server nicht die Fähigkeit einen Flash-Stream zu generieren. Da jedoch die Streams für die Webseite im Flash-Format produziert

werden sollen, wird ein zusätzlicher Streaming-Server benötigt der diese Aufgabe übernimmt.

Streaming-Server für die Flash-Streams

Der Wowza Media Server ist in der Lage die bestehenden MP3-Streams, die der Icecast Server schon zur Verfügung stellt, in Flash-Streams umzuwandeln und an einen integrierten Flash Player zu schicken. Da der Wowza Media Server auf der Programmiersprache Java beruht, ist es möglich am Server selber bei Bedarf Änderungen durchzuführen. So wäre es zum Beispiel denkbar, dass der Server nur anhand eines Codewortes einen Stream zu einem Flash Player übermittelt, welches einen hohen Vorteil in der Sicherheit der Streams bedeuten würde. Dies soll aber nur als Hinweis und Gedankenanstoß dienen und nicht Aufgabe des hier entstehenden Systems sein.

Ein möglicher Lösungsweg um die Audio Streams des intermedialen Redaktionssystems zu verteilen, wäre also die Kombination eines Icecast Servers mit einem Wowza Media Server. So könnte der Icecast Server die Basis des Streamingangebotes als MP3-Streams zur Verfügung stellen, die so bereits an die Media Player wie den Winamp ausgeliefert werden können. Der Wowza Media Server könnte diese Streams, in das für den im Portal integrierten Flash Player benötigte Format, umwandeln und zur Verfügung stellen.

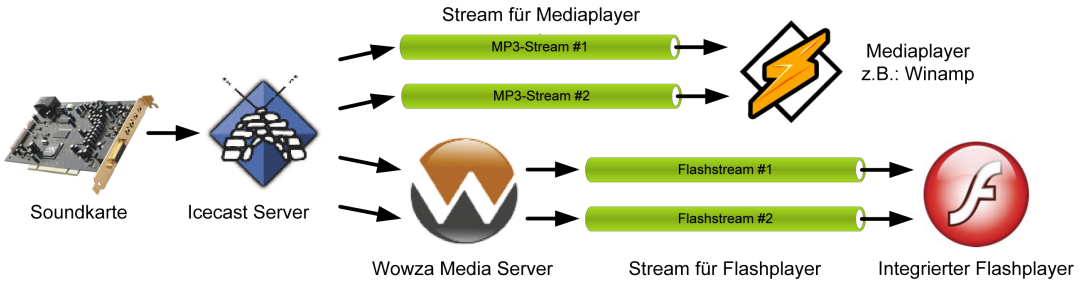


Abbildung 20: Streaming von einer Soundkarte mittels dem Icecast Server und dem Wowza Server

6.1.2 Encoder für die Audio-Streams

Um die Audio-Streams auch in verschiedenen Qualitäten für einen Streaming-Server bereitstellen zu können, benötigt eine Streaminglösung einen oder mehrere Encoder, die die analogen Signale von einer oder mehreren Soundquellen abgreifen, in die gewünschten Qualitäten und gewünschte Formate umwandeln und zu einem Streaming-Server senden können. Dieser Arbeitsschritt kann von externen Hardware-Encodern vollzogen werden, was einen großer Vorteil für die Stabilität und Ausfallsicherheit der Streaminglösung be-

deuten würde, jedoch auch extra Kosten für die Anschaffung der Hardware-Encoder verursacht.

Der Einsatz von Software-Encodern, auch Sourceclients genannt, bietet Vorteile was die Kosten in der Anschaffung betrifft, da es kostenfreie Encoder für verschiedenste Plattformen gibt, und in der Flexibilität bei einer möglichen Erweiterung des Streamingangebotes.

Für die erste Etappe, oder auch als endgültige Lösung, für das intermediale Redaktionssystem, bietet sich der Live Encoder DarkIce an. Dieser Encoder kann Audiosignale von einer Soundkarte abgreifen und unter anderem im MP3-Format an einen Icecast Server schicken. Da dieser Encoder unter dem Betriebssystem Debian/Ubuntu funktioniert, kann er auf dem selben System wie der Streaming-Server installiert und betrieben werden. Durch die Möglichkeit den Encoder auf dem System automatisch starten und zu dem Streaming-Server verbinden zu lassen, würde sich mit einer solchen Lösung die Möglichkeit bieten ein automatisches System zu kreieren.

6.1.3 Der Audio Flash Player für die Portalseite

Um einen Flash Player für Audio-Streams in eine Webseite integrieren und betreiben zu können, reicht ein durch die Encoder produzierter und durch den Icecast Server versendeter MP3-Stream völlig aus. Es gibt diverse Flash Player die einen so produzierten Stream abspielen können. Jedoch ist diese Lösung nur für das Versenden von Audio-Streams geeignet und die Einbindung von eventuellen Zusatzinformationen innerhalb des Flash Players nur über eine externe XML-Schnittstelle möglich.

Um einen Flash Player für die Bereitstellung von Audio- und Videodaten betreiben zu können, ist es nötig einen Flash-Stream zu produzieren. Ein solcher Flash-Stream bietet im Bereich des Audio-Streaming den Vorteil Zusatzinformationen, zum Beispiel den aktuellen Titel oder die Uhrzeit, in den Metadaten des Streams mit zum Player übertragen zu können. So erreichen den Benutzer auch auf der Webseite die gewünschten Zusatzinformationen in nahezu Echtzeit.

Für die Produktion eines solchen Audio-Flash-Streams gibt es verschiedene Vorgehensweisen, sie ist jedoch mit einem bereits vorhandenem MP3-Stream, den ein Icecast Server liefert, sehr viel einfacher.

Eine sehr einfache, aber auch mit laufenden Kosten verbundene, Lösung wäre es den bereits vorhandenen MP3-Stream an einen externen Provider zu leiten der dann über seine eigenen Server den Flash-Stream an einen Flash Player sendet. Diese Lösung hätte den Vorteil das die Erstellung, Organisation und Verwaltung eines eigenen Flash-Streaming-Servers entfällt, aber auch den Nachteil, dass man auf die Dienste, Möglichkeiten und

Reaktionszeiten des externen Anbieters angewiesen ist.

Ein anderer Lösungsweg wäre es, einen eigenen Flash-Streaming Server zu betreiben. Dieser Weg würde nur einmalig Kosten für die benötigte Server Software verursachen. Sie wäre wesentlich flexibler in der Anpassung an die Bedürfnisse des Streamingangebotes und könnte zusätzlich auf einem bereits bestehendem Streaming-System aufgesetzt werden. Ein geeigneter Flash Media Server wäre zum Beispiel der „Wowza Media Server“. Dieser Server ist für verschiedene Betriebssysteme erhältlich und ist in der Lage Video- und Audioinhalte als Live- und On-Demand-Stream an einen Flash Player zu übertragen.

6.1.4 Die Metadaten der Audio-Streams

Auch hier bietet der Icecast Server wieder die notwendigen Fähigkeiten. Der Icecast Server ist in der Lage die Metadaten in nahezu Echtzeit ändern zu können. Er bietet sogar von Haus aus die Möglichkeit jedem von ihm übertragenen Stream feste Metadaten zu geben. Zusätzlich besitzt er die Möglichkeit, über ein kleines integriertes Webinterface, die Metadaten von zum Beispiel einem Redakteur nachträglich aktualisieren lassen zu können.

Macht man sich diese Funktionen und Fähigkeiten des Icecast zu nutzen, ist es mit dem Einsatz von Shell- und PHP-Skripten möglich ein System zu erschaffen bei dem automatisch Daten, aus zum Beispiel einer XML-Datei oder einer Datenbank, ausgelesen und in die Metadaten eines Streams geschrieben werden. Es empfiehlt sich daher ein System zu entwickeln welches automatisch, in einem bestimmten zeitlichem Intervall, Informationen aus einer extra dafür abgelegten XML-Datei mit den aktuellen Metadaten im Stream vergleicht und diese bei einer Veränderung automatisch aktualisiert.

Denkbar ist bei einem solchen Workaround ebenfalls die Anbindung einer DABiS Schnittstelle. Dabei werden die Informationen zu einem aktuellen Titel aus einer XML-Datei mit den Informationen aus einer DABiS-Schnittstelle abgeglichen und bei Bedarf aktualisiert. Mit einem solchem Workaround ist es möglich Informationen des aktuellen Titels aus einer DABiS-Schnittstelle zu gewinnen und Zusatzinformationen, von zum Beispiel den Moderatoren, mit in die Metadaten der Streams zu schreiben.

6.1.5 Konzept einer automatischen Aktualisierung der Metadaten

Im folgendem Abschnitt wird ein möglicher Workaround erläutert, welcher in der Lage ist die Metadaten eines Audio-Streamingangebotes automatisch zu aktualisieren. Dabei werden die Metadaten, die den aktuellen Inhalt beschreiben und von einer DABiS-Plattform geliefert werden, ausgelesen und über einen Icecast Streaming Server mit zu dem Benutzer übertragen.

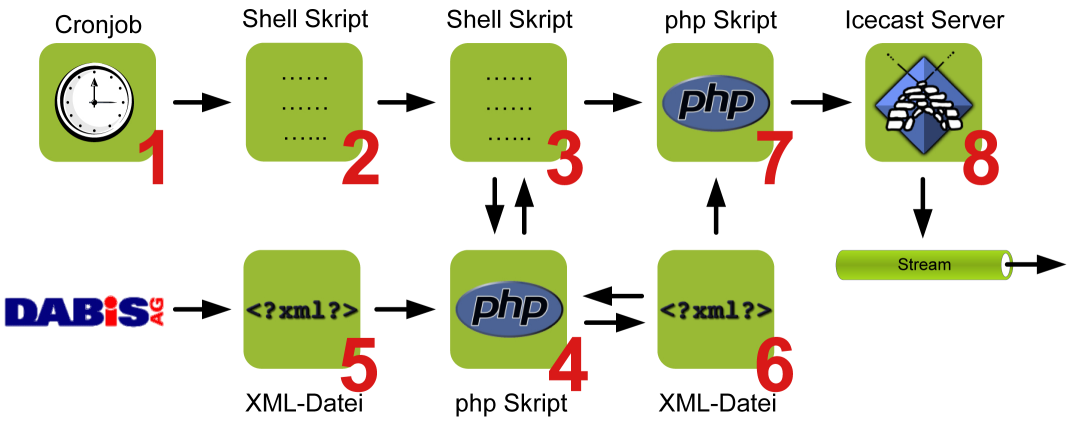


Abbildung 21: Workaround der Metadatenaktualisierung

Der Cronjob

Der erste Schritt für ein solches System wäre ein Cronjob. Der Cronjob dient dazu um beim Hochfahren des Rechners, oder bei einem Fehler des Workaround, den gesamten Prozess zu starten. Dabei versucht der Cronjob regelmäßig, innerhalb eines bestimmten Intervalls den Prozess auszuführen. Er schafft dies aber nur wenn der Prozess nicht läuft, also er noch nicht gestartet oder er durch einen Fehlerfall geschlossen wurde. Das kleinste einzustellende Intervall bei einem Cronjob ist eine Minute. So wäre bei einem Fehlerfall, mit anschließendem erfolgreichen Neustart, der Workaround maximal eine Minute nicht im Stande die Metadaten innerhalb der Streams zu aktualisieren.

Das Steuerskript für das Intervall

Das erste von zwei Shell Skripten in diesem Workaround dient dazu um einen zeitlichen Rahmen zu schaffen, indem die Aktualisierung der Metadaten durchgeführt wird. Dieses Skript hilft dabei das Intervall, in welchem die Metadaten aktualisiert werden, zu steuern und den Vorgang in Gang zu setzen. So könnte diese Skript zum Beispiel jede Sekunde den Vorgang der Metadatenaktualisierung starten.

Das Shell Skript für den Vergleich

Das zweite Skript in diesem Workaround dient dem Vergleich der alten und der neuen Metadaten. Hier wird entschieden ob eine Aktualisierung der Metadaten vorgenommen werden muss, oder nicht. Dabei stellt dieses Skript fest ob die Metadaten die von der DABiS-Schnittstelle zur Verfügung gestellt werden, mit den aktuellen Metadaten die sich im Stream befinden übereinstimmen. Ist dies nicht der Fall, sind die Metadaten im Stream veraltet und müssen erneuert werden, die Aktualisierung wird ausgelöst.

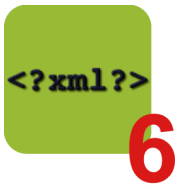
Das PHP Skript für den Vergleich

Dieses PHP-Skript führt den Vergleich der Metadaten von der DABiS Schnittstelle und einer eigenen XML-Datei durch. Sind die ausgelesenen Metadaten nicht identisch, wird die Aktualisierung ausgelöst. Wird bei diesem Arbeitsschritt ein Fehler festgestellt, können die Metadaten mit einem Standardwert versehen werden, wodurch eine veraltete oder fehlerhafte Anzeige in den Endgeräten vermieden wird. Ein Fehlerfall könnte zum Beispiel beim lesen der Metadaten auftreten, wenn die Quellen nicht erreichbar sind. Ebenfalls könnte man einen Zeitstempel mit in diese Struktur mit einbringen. Ist eine Beschreibung innerhalb der Metadaten länger als 10min aktiv, liegt wahrscheinlich ein Fehler von der DABiS-Schnittstelle vor.

Die XML-Datei mit den originalen Metadaten

In dieser XML-Datei stehen die aktuellen Informationen zu dem Inhalt des Programmes. Das DABiS-System speist diese XML-Datei mit Informationen wie dem aktuellen Titel und den folgenden Titeln, kann aber auch Zusatzinformationen wie den Namen des Moderators, das Cover des Album oder die Länge des Titels mit dieser Schnittstelle zur Verfügung stellen. Eine Auflistung von XML-Dateien die DABiS zur Verfügung stellen kann, ist im Anhang ersichtlich.

Die XML-Datei mit den aktuellen Metadaten



Innerhalb dieser XML-Schnittstelle werden alle Informationen, die letztendlich innerhalb der Streams als Metadaten fungieren, aufbewahrt. Durch den Vergleich und der ständigen Aktualisierung des Inhaltes mit dem Inhalt der XML-Schnittstelle, die DABiS zur Verfügung stellt, stehen in dieser XML-Datei immer die aktuellen Metadaten die gerade zum Benutzer übertragen werden. Durch den Einsatz von Zusatzanwendungen könnten an dieser Stelle noch Zusatzinformationen die zum Beispiel der Moderator auf der Webseite eintippt oder Zusatzinformationen die DABiS nicht liefern kann, mit Einzug in die Metadaten der Streams halten.

Das Skript für die Aktualisierung

Dieses PHP-Skript aktualisiert die Metadaten der Streams. Wird eine Änderung der Metadaten zwischen der DABiS-Schnittstelle und der Schnittstelle die die aktuellen Metadaten hält festgestellt, ruft dieses Skript die neuen Metadaten ab und übergibt sie an den Streaming-Server der die Aktualisierung durchführt.

Der Streaming-Server

Wird von dem Workaround die Notwendigkeit einer Aktualisierung festgestellt, erhält der Icecast Server die neuen Metadaten von dem PHP-Aktualisierungsskript und fügt diese Daten schließlich in die Streams ein und überträgt sie mit zum Benutzer.

6.1.6 Mögliche DABiS-Schnittstellen

DABiS ist in der Lage verschiedene XML-Outputs als Schnittstelle für die Metadaten zur Verfügung zu stellen. Je nachdem welche Module für das DABiS System installiert sind, gibt es die Möglichkeit eine XML-Schnittstelle mit allgemeinen Metadaten, Metadaten mit Platzhaltern die Zusatzinformationen enthalten und sogar allgemeine Metadaten mit Zusatzinformationen und Bilder zu generieren.

Im einfachsten Beispiel könnte DABiS also eine Schnittstelle zur Verfügung stellen, in-der der aktuelle Titel und X folgende Titel aufgeführt sind. Eine weitere Möglichkeit wäre es eine Schnittstelle zu generieren die neben dem aktuellen Titel und den folgenden Titeln gewissen Platzhalter enthalten. Diese Platzhalter würde das DABiS System automatisch mit den Inhalten füllen. Dabei ist die Struktur der Zusatzinhalte (wie zum Beispiel Startzeit, Länge des Titels, der Moderator) frei wählbar.

Eine weitere Möglichkeit wäre die Generierung einer Schnittstelle die völlig frei definierbar ist. Dabei kann mittels einer Programmiersprache (NXC - Not eXactly C) eine Schnittstelle erstellt werden, die eigens auf die benötigten Bedürfnisse angepasst ist.

Im Anhang befinden sich drei Beispiel-XML-Dateien, „Allgemeine Metadaten“, „Current/Next Titel inkl. Zusatzinformationen“ und „Beispiel eines Exports mittels NXC“, die die eben erwähnten Schnittstellen genauer beschreiben.

6.2 Konzept der Streaminglösung für die Video-Streams

In den folgenden Kapiteln werden mögliche Konzepte vorgestellt, durch die es möglich ist eine On-Demand Streaminglösung für die Video Clips auf der Portalseite zu gewährleisten. Ebenfalls finden sich hier Konzepte wodurch das Portal in der Lage ist, eine Live TV-Signal des Senders MW*DIGITAL* für die Benutzer zur Verfügung zu stellen.

6.2.1 Bereitstellen der Video Clips

Um die Präsentation von Video Clips auf dem resultierendem Portal zu bewerkstelligen, gibt es zwei mögliche Lösungsansätze, wobei nur eine den zuvor aufgestellten Anforderungen gerecht werden kann. Beide Ansätze nutzen die Möglichkeit die Videos mittels eines Adobe Flash Player auf dem Portal darzustellen. Ein Player auf der Basis von Adobe Flash bietet sich für die Präsentation von Videodaten an, da Adobe Flash eine sehr beliebte und

sehr weit verbreitete Erweiterung für die meisten Browser ist. Somit ist der Benutzer nicht gezwungen ein extra Plugin oder spezielle Software zu installieren um die Inhalte benutzen zu können. Ein weiterer Vorteil ist, dass ein Player der auf dieser Technik beruht in seinem Aussehen und seinem Funktionsumfang verändert werden kann.

6.2.2 Video Clips ohne einen Streaming-Server bereitstellen

Ein einfacher Weg die Video Clips auf dem Portal bereitzustellen, ist die Wandlung der Videodaten in ein Format, welches ein auf der Webseite integrierter Flash Player einfach abspielen kann. Adobe Flash entwickelte für diesen Zweck ein eigenes Videocontainerformat mit dem Namen Flash Video (.flv). Videodateien in diesem Format können dabei ohne einen Streaming-Server verbreitet werden, sie werden über das HTTP-Protokoll direkt von einem Flash Player geladen und wiedergegeben. Jedoch müssen die Videodaten, die man betrachten möchte, erst in einen Puffer geladen werden. Somit ist das springen zu einer beliebigen Stelle innerhalb des Videos erst dann möglich, wenn diese Stelle in den Puffer geladen wurde.

Aufgrund der vorher aufgestellten Anforderungen kommt dieser Ansatz nicht in Frage, stellt aber einen durchaus akzeptablen Kompromiss dar.

6.2.3 Video Clips mit einem Streaming-Server bereitstellen

Ein weiterer möglicher Workaround wäre, die Videodateien durch einen Flash Streaming-Server zu einem in der Webseite integrierten Flash Player zu übertragen. Dabei können die Videodateien in verschiedenen Formaten wie zum Beispiel dem Flash Video (.flv) oder dem QuickTime Container Format MP4 (.mp4, .f4v, .mov, .m4v, .mp4a, .3gp, und .3g2) auf dem Server vorliegen. Diese Dateien werden durch den Streaming-Server über das RTP/RTSP Protokoll an den Benutzer übertragen. (WOWZA ONLINE 2)

Ein großer Vorteil eines Workaround unter Zuhilfenahme eines Flash Media Server ist es, dass die Videodateien nicht erst vom Benutzer vorgeladen werden müssen und somit ein direktes springen an einen beliebigen Punkt innerhalb des Videos möglich ist. Dieser Vorteil macht sich extrem bei großen und langen Videos bemerkbar, da so unnötige Wartezeiten durch das Herunterladen vermieden werden.

Ein Flash Media Server der in der Lage ist die Videoinhalte auf diese Weise zur Verfügung zu stellen, ist der Wowza Media Server. Dabei bietet er nach der Installation gut erklärte Beispiele und liefert die nötigen Quellen um einen Flash Player nach eigenen Vorstellungen erstellen zu können.

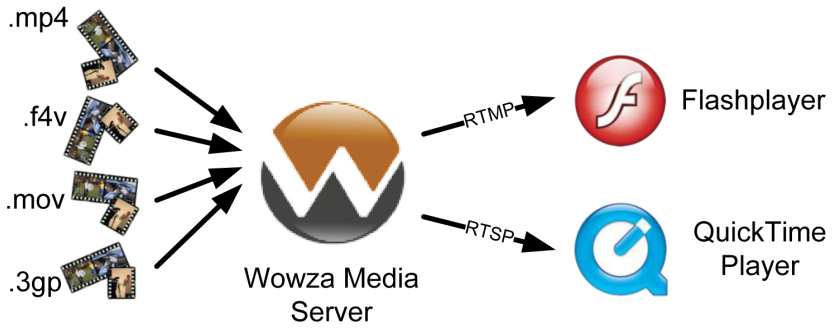


Abbildung 22: Streaming von Videodateien mittels des Wowza Servers

Wandlung und Bereitstellung der Video Clips

Um die Video Clips dem Benutzer zur Verfügung stellen zu können, müssen die Videodateien im Vorfeld bei beiden Workarounds in ein passendes Format gewandelt werden. Um diesen Prozess für die Redakteure zu vereinfachen, könnte ein System geschaffen werden welches die originalen Videodaten automatisch in das geforderte Format wandelt und anschließend auf dem Portal zur Verfügung stellt.

Ein möglicher Workaround wäre mit der kostenfreien Transkodierungssoftware FFmpeg zu realisieren. Dabei wandelt FFmpeg die von den Redakteuren hochgeladen Videos automatisch in das entsprechende Format um. Denkbar wäre ein System, bei dem in einem bestimmten Intervall geprüft wird ob ein Redakteur ein Video hochgeladen hat. Ist dies der Fall transkodiert FFmpeg dieses Video in das geforderte Format und stellt es dem Portal zur Verfügung.

6.2.4 Streaming des TV-Signals

Ein möglicher Lösungsweg für das Streamen des Live TV-Bildes wäre der Einsatz eines Live-Encoders wie den Adobe Flash Media Encoder (AFME) und dem Einsatz des Flash Streaming-Servers Wowza Media Server. Simultan zu der Bereitstellung der Video Clips, könnte so ein auf der Webseite integrierter Flash Player das Livebild zur Verfügung stellen.

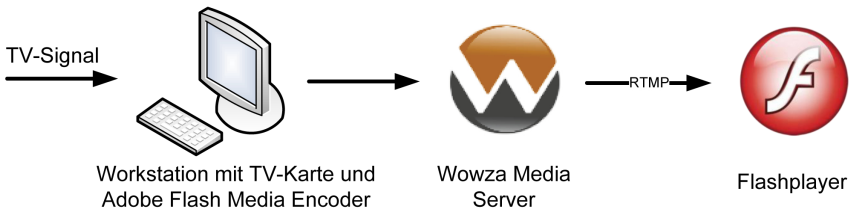


Abbildung 23: Streaming mit einem Adobe Flash Media Encoder über den Wowza Media Server.

Bei diesem Lösungsansatz könnte der Adobe Flash Media Encoder das Signal einer Videoechtzeitquelle, zum Beispiel einer TV- oder Videokarte, an den Wowza Media Server übermitteln. Dieser Lösungsansatz würde jedoch den Einsatz eines extra dafür bereitgestellten PC's voraussetzen, da der Adobe Flash Media Encoder nur für den Einsatz unter dem Betriebssystem Microsoft Windows zur Verfügung steht.

7 Prototypische Umsetzung der Streaminglösung

In den folgenden Kapiteln wird die Inbetriebnahme einer solchen Sreaminglösung anhand einer Testumsetzung veranschaulicht. Dabei wird mit den Streaming-Servern „Icecast Server“ und „Wowza Media Server“ und dem Sourceclient „DarkIce“ eine Umgebung für das Audio-Live-Streaming geschaffen, welche in der Lage ist das Eingangssignal einer Soundkarte mittels MP3-Streams und Flash-Streams zur Verfügung zu stellen und die Metadaten innerhalb der Streams zu aktualisieren.

Auf die Testumsetzung einer Video-On-Demand und einer Live-Videostreaminglösung wird in dieser Arbeit verzichtet, da der Wowza Flash Media Server nach der Installation sehr gut erklärte Beispiele für beide Lösungen enthält.

Während der folgenden Kapitel wird ebenfalls eine Datei- und Verzeichnisstruktur für den Workaround einer Streaminglösung entwickelt, die den Administratoren die Verwaltung und Pflege erleichtern soll.

7.1 Die Streaming-Server

In den folgenden Kapiteln werden die für das Audio- und Video-Streaming erforderlichen Streaming-Server „Icecast Server“ und „Wowza Media Server“ unter einem Ubuntu Betriebssystem installiert und konfiguriert.

7.1.1 Icecast Server installieren

Am Anfang müssen natürlich die Quellen für den Icecast Server vorliegen. Da die Quellen für den Icecast Server in dem aktuellen Repositorium von Ubuntu zu finden sind, ist das Herunterladen und die Installation bei einer bestehenden Internetverbindung denkbar einfach.

```
sudo apt-get install icecast2
```

Mit Hilfe dieses Kommandos, werden alle benötigten Dateien und Pakete automatisch heruntergeladen und installiert.

Der Icecast Server wird über eine zentrale Konfigurationsdatei konfiguriert und verwaltet. Diese Konfigurationsdatei liegt standardmäßig unter dem Pfad */etc/ icecast2/icecast.xml*. Allerdings hat es sich bewährt einen eigenen Benutzer für das Thema Streaming auf dem Server einzurichten und alle benötigten Konfigurationsdateien (für die Streaming-Server, für die Encoder, für die Transcoder) in dessen Home-Verzeichnis abzulegen. Ebenfalls hat es sich bewährt, dass der Icecast Server unter diesem Benutzer läuft und von ihm gestartet wird.

▸ /etc/icecast2/icecast.xml

Hier liegt standardmässig nach der Installation eine Konfigurationsdatei für den Icecast Server.

Nehmen wir an der Benutzer, der für das Streaming zuständig sein soll, heißt „streaming“. Sein Home-Verzeichnis müsste dann unter dem Pfad „/home/streaming“ zu finden sein. Um die Gebiete Streaming, Encoding, Metadaten und andere logisch voneinander zu trennen und die Verzeichnisstruktur übersichtlich zu halten, sollte sich in dem Home-Verzeichnis das Unterverzeichnis „icecast“ befinden. Hier befinden sich alle Dateien die für den Betrieb und die Konfiguration des Icecast Server benötigt werden.

▸ /home/streaming/icecast

In diesem Verzeichnis liegen alle notwendigen Konfigurationsdateien und Skripte die zur Ausführung des Icecast Server benötigt werden.

Um den Benutzer „streaming“, sein Home-Verzeichnis und die Unterverzeichnisse verwenden zu können, muss dieser Benutzer und die Verzeichnisstruktur zuerst angelegt werden.

▸ sudo adduser streaming

Fügt den Benutzer „streaming“ hinzu und erstellt dessen Home-Verzeichnis.

▸ sudo mkdir /home/streaming/icecast

Legt das Verzeichnis „icecast“ in dem Home-Verzeichnis des Benutzers „streaming“ an.

Um die Standard-Konfigurationsdatei nutzen zu können, muss diese in die neue Verzeichnisstruktur eingepflegt werden.

▸ sudo mv /etc/icecast2/icecast.xml /home/streaming/icecast/

Verschiebt die Standard-Konfigurationsdatei in die neue Verzeichnisstruktur.

Der Icecast Server verfügt über ein Logging, damit evtl. auftretende Fehler leichter erkannt und behoben werden können. Um dieses Logging benutzen zu können, benötigt der Icecast Server einen Ort um seine Log-Dateien abzuspeichern. Um die neue Verzeichnisstruktur zu nutzen, wird ein Verzeichnis „logfiles“ in dem Verzeichnis „icecast“ des Benutzers „streaming“ benötigt. Hier liegen später die beiden Log-Dateien „access.log“ und „error.log“.

```
sudo mkdir /home/streaming/icecast/logfiles
```

Legt ein neues Verzeichnis für die Log-Dateien an.

Der Icecast Server verfügt über ein eigenes kleines Webinterface, welches zur Kontrolle der Streams dient. Um dieses mit dem User „streaming“ nutzen zu können, müssen die bereits vorhandenen Verzeichnisse „web“ und „admin“ ebenfalls in die neue Verzeichnisstruktur eingepflegt werden.

```
sudo mv -R /etc/icecast2/web /home/streaming/icecast/
```

```
sudo mv -R /etc/icecast2/admin /home/streaming/icecast/
```

Verschiebt das komplette Verzeichnis „web“ und „admin“ in die neue Verzeichnisstruktur.

Damit der Icecast Server auch auf die neu erstellte Verzeichnisstruktur zugreifen kann, müssen für jedes Verzeichnis die Rechte, Besitzer und die Gruppenzugehörigkeit angepasst werden.

```
sudo chgrp -R streaming /home/streaming/icecast
```

```
sudo chown -R streaming /home/streaming/icecast
```

Mit Hilfe dieser beiden Kommandos, werden die Gruppenzugehörigkeit und der Besitzer des Verzeichnisses „streaming“ und dessen Unterverzeichnisse auf die Gruppe „streaming“ und den Benutzer „streaming“ gesetzt.

Konfigurationdatei und Besitzer bekannt machen

Um den Icecast Server für seinen ersten Start vorzubereiten, muss ihm noch mitgeteilt werden, welche Konfigurationsdatei er benutzen und unter welchem Benutzer er in Zukunft laufen soll. Dazu muss die Datei, die unter dem Pfad */etc/default/icecast2* liegt, angepasst werden.

In dieser Datei müssen die Werte für den Pfad zu der Konfigurationsdatei, die User-ID und Gruppen-ID angepasst werden. Ist dies geschehen, muss der Parameter „ENABLE“ auf „true“ gesetzt werden um diese Konfiguration für den Icecast Server zu aktivieren.

/etc/default/icecast2

```
CONFIGFILE="'/home/streaming/icecast/icecast.xml"'
USERID=streaming
GROUPID=streaming
ENABLE=true
```

Grundkonfiguration des Icecast Server

Um den Icecast Server den Zugriff auf die Log-Dateien und des Webinterfaces zu ermöglichen, müssen in der Konfigurationsdatei die Namen und die Positionen der Log-Dateien, sowie die Position der beiden Webinterfaceverzeichnisse „admin“ und „web“ konfiguriert werden.

/home/streaming/icecast/icecast.xml

```
<logdir>/home/streaming/icecast/logfiles</logdir>
<accesslog>access.log</accesslog>
<errorlog>error.log</errorlog>
<webroot>/home/streaming/icecast/web</webroot>
<adminroot>/home/steraming/icecast/admin</adminroot>
```

Starten des Icecast Server

Nachdem diese notwendigen Grundeinstellungen getroffen wurden, die Konfigurationsdatei an ihrem Platz ist und die Log- und Web-Verzeichnisse eingerichtet sind, ist der Icecast Server für seinen ersten Start bereit. Mit dem Kommando „/etc/init.d/icecast2 start“ wird der Server im Vordergrund gestartet, möchte man ihn im Hintergrund starten, benutzt man „/etc/init.d/icecast2 start &“.

/etc/init.d/icecast2 start

Startet den Icecast Server im Vordergrund

/etc/init.d/icecast2 start &

Startet den Icecast Server im Hintergrund

Um den erfolgreichen Start des Web-Interfaces zu testen, muss in einem Browser die URL „http://SERVERS_IP:8000“ eingegeben werden.

7.1.2 Icecast Server konfigurieren

Der Icecast Server wird über seine zentrale Konfigurationsdatei gesteuert. In dieser Datei werden die Parameter wie „Wer darf alles hören?“, „Wieviele Hörer dürfen connecten?“, „Von wo wird der Stream abgegriffen?“, „Wo wird der Stream hin gesendet?“ verwaltet.

Die Konfigurationsdatei des Icecast Server ist eine XML-Datei und obliegt somit auch den Regeln und Vorschriften dieser Auszeichnungssprache. In dem Wurzelelement „icecast“ gibt es 10 weitere Hauptelemente, die die wichtigsten Bereiche beschreiben und weitere spezifische Parameter einschließen.

Der Icecast Server kann unter dem Betriebssystem Ubuntu mehrfach gestartet werden. So ist die Verwendung mehrere Icecast Server auf einem Rechner gewährleistet. Dabei können mehrere Konfigurationsdateien mit unterschiedlichen Konfigurationen in der Verzeichnisstruktur hinterlegt werden.

Um den Icecast Server für den Testbetrieb zu konfigurieren, müssen in dieser Konfigurationsdatei einige Einstellungen vorgenommen werden. Wichtige grundsätzliche Parameter sind „admin-password“ durch welches ein Sourceclient auf den Streaming-Server verbinden darf und der „port“, der angibt unter welchem Port der Streaming-Server erreichbar ist. Mit diesen zwei grundsätzlichen Einstellungen kann der Icecast Server bereits gestartet und mit einem Sourceclient verwendet werden. Jedoch sind noch eine Vielzahl an Konfigurations- und Optimierungseinstellungen zu treffen um einen optimalen Betrieb zu gewährleisten.

Im nachfolgendem Kapitel werden die Parameter der Konfigurationsdatei einmal genauer beschrieben und erklärt.

7.1.3 Wichtige Parameter in der „icecast.xml“.

Grundaufbau der XML-Datei:

```
<icecast>
    <limits></limits>
    <authentication></authentication>
    <directory></directory>
    <hostname></hostname>
    <listen-socket></listen-socket>
    <mount></mount>
    <fileserve>1</fileserve>
    <paths></paths>
    <logging></logging>
    <security></security>
</icecast>
```

Das Element „limits“

In dem Element „limits“ werden die Beschränkungen für die Anzahl der Streams und Quellen festgelegt. Diese Werte müssen je nach Geschwindigkeit der Anbindung des Servers und der angebotenen Qualität des Streams gewählt werden.

Beispielinhalt:

```
<clients>100</clients>
<sources>2</sources>
<threadpool>5</threadpool>
<queue-size>102400</queue-size>
<client-timeout>30</client-timeout>
<header-timeout>15</header-timeout>
<source-timeout>10</source-timeout>
<burst-on-connect>1</burst-on-connect>
<burst-size>65535</burst-size>
```

clients: Setzt den Wert der maximal erlaubten Anzahl an Zuhörern die sich gleichzeitig auf den Icecast verbinden können.

sources: Setzt den Wert der maximal gleichzeitig verbundenen Quellen des Icecast.

threadpool: Setzt den Wert für die Anzahl der Prozesse, die gestartet werden um die Anfragen zu verarbeiten.

queue-size: Die Größe einer internen Warteschlange in Bytes für die Zuhörer. Wird der Wert von einem Hörer überschritten, wird er vom Stream getrennt.

client-timeout: Die Zeit in Sekunden, nachdem ein Hörer bei einem Timeout getrennt wird.

header-timeout: Die Zeit in Sekunden die der Server auf einen Request wartet, nachdem ein Client sich verbunden hat.

source-timeout: Die Zeit in Sekunden, nachdem eine Quelle vom Server getrennt wird wenn sie keine Daten mehr liefert.

burst-on-connect: Bei aktiviertem burst-on-connect wird dem gerade erst verbundenem Klienten ein Paket an Audiodaten geschickt. Diese Paket hat die Aufgabe den Puffer des Players vom Klienten zu füllen. Somit startet die Wiedergabe des Streams für den Klienten spürbar eher, da die meisten Player erst dann anfangen abzuspielen wenn der Puffer gefüllt ist.

burst-size: Ist die Größe des Datenpaketes in Byte, welches bei einer Neuverbindung an den Klienten geschickt wird.

Das Element „authentication“

In dem Element „authentication“ werden die Zugriffsberechtigungen der Quellen, eines möglichen Relay und des Webinterface für den Icecast definiert.

Beispielinhalt:

```
<source-password>hackme</source-password>
<relay-user>relay</relay-user>
<relay-password>hackme</relay-password>
<admin-user>admin</admin-user>
<admin-password>hackme</admin-password>
```

source-password: Definiert ein Passwort, welches in den Quellen (z.b.: Ices, Encoder) angegeben werden muss, damit diese auf den Server verbinden dürfen.

relay-user: Der Name eines möglichen Relay-Benutzers, der über diesen Icecast seinen Stream versenden möchte.

relay-password: Das Passwort eines möglichen Relay-Benutzers, der über diesen Icecast seinen Stream versenden möchte.

admin-user: Der Name des Administrators für das Webinterface des Icecast Server.

admin-password: Das Passwort des Administrators für das Webinterface.

Das Element „directory“

In dem Element „directory“ werden die Einstellungen für die Icecast2 Yellow Page Directory Server vorgenommen. Diese Server stellen eine Art „Gelbe Seiten“ für die Hörer bereit, damit die einzelnen Icecast Streams und Server besser anhand der eingetragenen Metadaten gefunden werden können.

Beispielinhalt:

```
<yp-url-timeout>15</yp-url-timeout>
<yp-url>http://dir.xiph.org/cgi-bin/yp-cgi</yp-url>
```

yp-url-timeout: Die maximale Zeit die der Icecast Server auf eine Antwort von einem Yellow Page Server wartet.

yp-url: Die URL eines Yellow Page Servers.

Allgemeine Server Einstellungen

Die Elemente „hostname“, „listen-socket“, „port“, „bind-adress“ und „fileserve“ beeinhalten die allgemeinen Einstellungen für den Icecast Server. Hier wird unter anderem die Netzwerkeinstellungen konfiguriert.

Beispielinhalt:

```
<hostname>localhost</hostname>
<listen-socket>
    <port>8000</port>
    <!-- <bind-address>127.0.0.1</bind-address> -->
</listen-socket>
<fileserve>1</fileserve>
```

hostname: Der Name des Rechners unter dem er im Netzwerk erreichbar ist.

Das Element „listensocket“ beinhaltet die Konfiguration für die eingehenden Clientverbindungen

port: Der TCP Port für die eingehenden Client Verbindungen.

bind-adress: Hier kann eine optionale IP-Adresse angegeben werden. Falls der Server über mehrere IP-Adressen verfügt, wird dann ausschließlich diese vom Icecast benutzt. Wird der Bereich nicht aktiviert, lauscht der Icecast Server an allen IP-Adressen nach möglichen Hörern.

fileserve: Dieses Flag aktiviert den internen Fileserver des Icecast von dem „On-Demand“ Inhalte bezogen werden können. Die Inhalte können dann relativ zu dem unter „paths“ - „webroot“ angegebenen Pfad hinterlegt und abgerufen werden.

Das Element „mount“

In einem Element „mount“ werden alle notwendigen Einstellungen für einen Mountpoint vorgenommen. In diesem Element werden wichtige Parameter für das Verhalten des Streams beim Klienten definiert, zum Beispiel „Was soll passieren wenn der Stream voll ist?“, „Was soll passieren wenn der Stream abbricht?“, „Von welchem Typ ist der Stream?“ oder „Welche Bezeichnung hat der Stream?“

Beispielinhalt des Elements „mount“:

```
<mount-name>/example-complex.ogg</mount-name>
<username>othersource</username>
```

```
<password>hackmemore</password>
<max-listeners>1</max-listeners>
<max-listener-duration>3600</max-listener-duration>
<dump-file>/tmp/dump-example1.ogg</dump-file>
<intro>/intro.ogg</intro>
<fallback-mount>/example2.ogg</fallback-mount>
<fallback-override>1</fallback-override>
<fallback-when-full>1</fallback-when-full>
<public>1</public>
<stream-name>My audio stream</stream-name>
<stream-description>My audio description</stream-description>
<stream-url>http://some.place.com</stream-url>
<genre>classical</genre>
<bitrate>64</bitrate>
<type>application/ogg</type>
<hidden>1</hidden>
<burst-size>65536</burst-size>
<mp3-metadata-interval>4096</mp3-metadata-interval>
<authentication type="htpasswd">
        <option name="filename" value="myauth"/>
        <option name="allow_duplicate_users" value="0"/>
</authentication>
<on-connect>/home/icecast/bin/source-start</on-connect>
<on-disconnect>/home/icecast/bin/source-end</on-disconnect>
```

mount-name: Der Name des Mountpoints für den die Einstellungen gelten.

username: Der Name eines optionalen Benutzers, den eine Quelle benutzen muss um diesen Mountpoint zu benutzen.

password: Ein optionales Passwort welches eine Quelle vorweisen muss um diesen Mountpoint zu benutzen.

max-listeners: Eine optionale Anzahl von maximalen Benutzern die auf den Mounpoint connecten dürfen.

max-listener-duration: Eine optionale maximale Zeit die ein Benutzer mit dem Mountpoint verbunden sein darf.

dump-file: Eine optionale Datei in der der eine Kopie des Stream gespeichert werden kann.

intro: Hier kann eine optionale Datei angegeben werden, die für einen Klienten direkt nach dem verbinden abgespielt wird.

fallback-mount: Hier kann ein anderer Mountpoint angegeben werden, auf den die verbundenen Klienten zurückfallen, wenn der Mointpoint offline gehen sollte. Es zum Beispiel zu einer Störung bei der Verbindung zwischen der Quelle und dem Icecast kommt.

fallback-override: Ist dieser Parameter aktiviert, wird ein Client der während einer Störung auf einen „fallback-mount“ gefallen ist, nach der Störung wieder auf den Stream zurückgeholt.

fallback-when-full: Gibt an, dass Clients die sich bei schon vollem Stream verbinden wollen auf den „fallback-mount“ geleitet werden.

public: Definiert ob der Mountpoint in den Yellow Pages gelistet werden soll oder nicht.

stream-name: Dieser Parameter definiert einen Namen für den Stream der über den Mountpoint versendet wird, der jedoch von der Quelle überschrieben werden kann.

stream-description: Hier kann eine Beschreibung zum Stream eingetragen werden, die jedoch ebenfalls durch die Quelle überschrieben werden kann.

stream-url: In diesem Element kann eine Adresse zu einer Website angegeben werden.

genre: Dieser Parameter definiert das Genre des Streams, welches in den Yellow Pages als Suchkriterium dient.

bitrate: Definiert die Bitrate des Streams, dieser Wert kann jedoch von der Quelle überschrieben werden, sollte er abweichen.

type: Hier kann der MIME-TYPE des Streams der über den Mountpoint läuft definiert werden. Zum Beispiel MPEG, AAC, AACP oder OGG.

hidden: Definiert ob der Mountpoint auf der Statusseite des Webinterface vom Icecast erscheinen soll oder nicht.

burst-size: Eine optionale Größe des Burst-Datenpaketes in Bytes. Dieser Wert überschreibt den „burst-size“-Wert der im Element „limits“ definiert wurde.

mp3-metadata-interval: Der Parameter definiert das Zeitliche Intervall mit dem die Metadaten in dem Stream an den Klienten geschickt werden.

authentication: In diesem Element kann eine Authentifizierung für die Klienten aktiviert und konfiguriert werden. Demnach müssen sich dann die Klienten mit einem Namen und einem Passwort am Server anmelden um auf die Angebote zugreifen zu können.

on-connect: Mit Hilfe dieses Parameters, kann ein Skript für Unix/Linux Systeme definiert werden, welches ausgeführt wird wenn ein Client auf den Mountpoint verbindet.

on-disconnect: Simultan zu „on-connect“ kann hier ein Skript definiert werden, welches dann ausgeführt wird wenn der Client vom Mountpoint disconnected.

Das Element „path“

Das Element „path“ beinhaltet die benötigten Pfad-Einstellungen für den Icecast und das integrierte Webinterface.

Beispielinhalt:

```
<basedir>./</basedir>
<logdir>./logs</logdir>
<pidfile>./icecast.pid</pidfile>
<webroot>./web</webroot>
<adminroot>./admin</adminroot>
<alias source="/foo" dest="/bar"/>
```

basedir: Legt das Verzeichnis fest, indem der Icecast zu finden ist. Diese Einstellung ist unter Windows nicht verfügbar.

logdir: Definiert den Pfad zu dem Verzeichnis indem die Log-Dateien abgelegt werden.

pidfile: Diese Datei klärt den Icecast (und andere Prozesse) über den Status des Icecast Servers auf. Sie wird angelegt wenn der Icecast Server läuft.

webroot: In dem hier definiertem Verzeichnis befinden sich die notwendigen Dateien für das Webinterface und können auch mögliche On-Demand Inhalte hinterlegt werden.

adminroot: Hier wird der Pfad zu dem Verzeichnis, indem alle benötigten Dateien für die Administrator Seiten des Webinterfaces liegen, definiert.

alias: Durch ein Element „alias“ kann ein Mountpoint auf einen anderen umgeleitet werden.

Das Element „logging“

Das Element „logging“ beinhaltet die Namen der Log-Dateien in dem evtl. auftretende

Fehler und die Zugriffinformationen der Klienten gespeichert werden. Ebenfalls wird hier das Level des Logging (Debug, Info, Warn, Error) und der Name einer Datei definiert, inder alle gespielten Titel hinterlegt werden.

Beispielinhalt:

```
<accesslog>access.log</accesslog>
<errorlog>error.log</errorlog>
<playlistlog>playlist.log</playlistlog>
<loglevel>4</loglevel> <-- 4 Debug, 3 Info, 2 Warn, 1 Error -->
```

accesslog: Diese Element beinhaltet den Namen der Log-Datei inder sämtliche Zugriffe auf den Icecast gespeichert werden.

accesslog: Die hier angegebene Log-Datei beinhaltet alle Fehlermeldungen, Warnungen und Hinweise, abhängig vom Log-Level.

playlistlog: In der hier definierten Datei werden alle Titel gespeichert die der Icecast gespielt hat.

loglevel: Das „loglevel“ gibt an welche Informationen gelogt werden. (Loglevel 4 = Fehler, Warnungen, Infos und Debug, 3 = Fehler, Warnungen und Infos, 2 = Fehler und Warnungen und 1 = Fehler)

Das Element „security“

In diesem Element werden serverseitige Sicherheitseinstellungen vorgenommen. Hier wird unter anderem der Benutzer definiert, unter dem der Icecast laufen soll.

Beispielinhalt:

```
<chroot>0</chroot>
<changeowner>
    <user>nobody</user>
    <group>nogroup</group>
</changeowner>
```

chroot: Anhand dieser Option, wodurch der Icecast Server unter einem anderem Root-Verzeichnis läuft und somit nicht mehr auf Dateien außerhalb zugreifen darf, kann der Prozess des Icecast in eine Art Testumgebung geschickt werden.

changeowner: Mit den beiden Elementen „user“ und „group“ kann der Benutzer und die Benutzergruppe definiert werden, unter der der Icecast Server laufen soll.

7.1.4 Wowza Media Server installieren.

Die Quellen des Wowza Media Server sind nicht im aktuellen Repositorium für die Installation unter Ubuntu zu finden, jedoch bietet der Hersteller auf seiner Webseite die Installationsroutinen für verschiedene Betriebssystem an. Vor der Installation müssen also die Quellen des Wowza Media Servers unter www.wowzamedia.com heruntergeladen werden. Hier finden sich sich die neusten Versionen des Wowza Media Servers. Alle folgenden Schritte beziehen sich auf die Version Wowza Media Server Pro 1.7.2.

Um die bisher verwendete Verzeichnisstruktur nicht zu verletzen, wird der Media Server mit in diese integriert. Dazu wird ein Verzeichnis für den Wowza im Home-Verzeichnis des Users „streaming“ benötigt.

```
sudo mkdir /home/streaming/wowza
```

In diesem Verzeichnis wird zuerst einmal nur die Installationsroutine abgelegt, später werden sich hier die benötigten Konfigurationsdateien befinden.

```
sudo wget http://www.wowzamedia.com/downloads/WowzaMediaServer-1-7-2/WowzaMediaServerPro-1.7.2.deb.bin -P /home/streaming/wowza
```

Lädt die benötigten Pakete für die Installationsroutine des Wowza Media Server in das neu angelegte Verzeichnis herunter.

Für die Installation des Wowza Media Servers wird ein Lizenzschlüssel benötigt. Da Wowza ihren Media Server als vollwertige Testversion, mit der Beschränkung auf 10 Streams, anbietet, gibt es auch einen entsprechenden kostenlosen Lizenzschlüssel für die Installation. Der Lizenzschlüssel lässt sich einfach auf der Website anfordern.

Nach dem Download der Installationsroutine muss das Skript Ausführungsrechte erhalten, damit die Installation gestartet werden kann.

```
sudo chmod +x WowzaMediaServerPro-1.7.2.deb.bin
```

Gibt dem Installationskript benötigte Ausführungsrechte.

Jetzt kann die eigentliche Installation beginnen.

```
sudo ./WowzaMediaServerPro-1.7.2.deb.bin
```

Während des ersten Startes wird die Installation des Wowza Media Servers ausgeführt. Hier muss auch der per E-Mail zugeschickte Lizenzschlüssel eingegeben werden.

```
cd /usr/local/WowzaMediaServer/bin
```

```
sudo ./startup.sh
```

Der Wowza Media Server basiert auf der Programmiersprache Java, deshalb ist eine Java Umgebung (Runtime Enviroment, Developer Kit und ein Plugin) auf dem System notwendig.

```
apt-get install sun-java6-jre sun-java6-plugin sun-java6-jdk
```

Im Anschluss an die Installation und dem ersten Start sollte man kontrollieren ob sich der Wowza Media Server über den HTTP-Port 1935 meldet. Es sollte eine Meldung ähnlich „Wowza Media Server Pro10 1.7.2 build12107“ im Browser zu sehen sein.

7.2 Der Sourceclient

Um die für die Streaminglösung erforderlichen Umwandlungen der Dateien und Signale vornehmen zu können, wird im folgendem Kapitel die Installation und Konfiguration des Sourceclienten „DarkIce“ unter dem Betriebssystem Ubuntu veranschaulicht.

7.2.1 Den Sourceclient „DarkIce“ installieren

Den Sourceclient DarkIce gibt im aktuellen Repositorium von Ubuntu. Daher lässt er sich über die integrierten Paketmanager leicht installieren.

```
sudo apt-get install darkice
```

Holt und installiert den Sourceclienten „DarkIce“.

Nach erfolgreicher Installation befindet sich eine Konfigurationsdatei in dem Verzeichnis */usr/local/etc/darkice.cfg*. Um die vorgegebene Verzeichnisstruktur beizubehalten, wird diese Konfigurationsdatei in die schon bestehende Struktur eingepflegt. Dazu wird ein Verzeichnis benötigt, indem alle Konfigurationsdateien von möglichen Sourceclients hinterlegt werden können.

```
mkdir /home/streaming/sourceclients
```

Erstellt ein Verzeichnis mit dem Namen „sourceclients“ indem alle Konfigurationsdateien abgelegt werden.

```
mkdir /home/streaming/sourceclients/darkice
```

Erstellt ein Verzeichnis für die Konfigurationsdateien des DarkIce Sourceclient.

```
cp /usr/local/etc/darkice.cfg /home/streaming/sourceclients/darkice
```

Kopiert die Konfigurationsdatei des DarkIce an die in der Verzeichnisstruktur eingerichtete Stelle.

7.2.2 Den Sourceclient „DarkIce“ konfigurieren

Die Konfigurationsdatei des DarkIce ist in drei Teile aufgeteilt. Für die Einrichtung eines Live-Audio-Streams im MP3-Format von einer Soundkarte zu einem Icecast Server, müssen in den Teilen „input“ und „icecast2-0“ gewisse Veränderungen vorgenommen werden.

Als Input ist die Soundkarte zu wählen, von der das analoge Signal abgefangen werden soll. In dieser Testumgebung ist das die Soundkarte die unter „/dev/dsp1“ zu erreichen ist. Damit der Sourceclient ein MP3-Signal an den Streaming-Server sendet, muss als Format in dem Teil „icecast2-0“ „mp3“ angegeben werden.

Damit DarkIce auf den schon installierten und konfigurierten Streaming-Server Icecast zugreifen kann und das auch darf, müssen in dem Teil „icecast2-0“ die IP und der Port auf der der Icecast Server zu erreichen ist und das Admin-Passwort angegeben werden. Damit der Streaming-Server weiß um welchen Stream es sich handelt, muss hier zusätzlich noch der Name des Streams angegeben werden.

Beispielinhalt:

```
[general]
duration = 0
bufferSecs = 5

[input]
device = /dev/dsp1
sampleRate = 48000
bitsPerSample = 16
channel = 2

[icecast2-0]
```

```
bitrateMode = cbr
format = mp3
bitrate = 128
quality = 1.0
server = 192.168.1.33
port = 8000
password = letsstream
mountPoint = stream_128
name =
description =
url =
genre =
public = yes
```

Start des DarkIce Sourceclient

Um den DarkIce zu starten, muss lediglich der Befehl „darkice“ eingegeben werden. Jedoch führt der DarkIce dann die Standard-Konfigurationsdatei aus und nicht die aus der angelegten Verzeichnisstruktur. Um die Konfigurationsdatei aus der Verzeichnisstruktur zu verwenden, muss dem Befehl „darkice“ mitgeteilt werden welche Konfiguration er verwenden soll. Mit dem Befehl *darkice -c Pfad zur Konfigurationsdatei* startet der DarkIce mit den richtigen Einstellungen.

Nach der Durchführung der bisher erwähnten Arbeitsschritte, sollte das System bereits in der Lage sein einen MP3-Basis Stream von der Soundkarte über den DarkIce Sourceclient an den Icecast Server zu schicken, der den Stream dann an einen Media Player, wie zum Beispiel den Winamp, schickt. Im folgendem Kapitel wird die Erstellung eines Flash Media Stream anhand eines so entstanden MP3-Streams näher erklärt.

7.3 Flash-Audio-Stream im Wowza Media Server einrichten.

Der Wowza Media Server bietet nach der Installation bereits einige fertige und gut erklärte Beispiele. Unter Anderem ein Beispiel mit dem Namen „SHOUTcast“ für die Einbindung eines Flash-Streams in einen Flash Player. Um mit folgender Beschreibung einen MP3-Stream von einem Icecast Server über einen Wowza Media Server als Flash-Stream zur Verfügung stellen zu können, wird zuerst ein funktionierender Icecast Server benötigt, der ein MP3-Stream ausliefert.

Um einen solchen Stream erstellen zu können, muss zuerst das SHOUTcast Beispiel über folgendem Wege auf dem Server installiert werden.

```
sudo ./install.sh
```

Der Wowza Media Server liefert zu dem SHOUTcast Beispiel gleich die passenden AS2 und AS3 Flash Player mit. Die zwei Player sind in dem Verzeichnis „SHOUTcast“ zu finden und können einfach in ein Verzeichnis, auf das ein Webserver Zugriff hat, kopiert werden.

In dieser Testumgebung hat der User „streaming“ ein Verzeichnis „public_html“ auf den der Apache Webserver Zugriff hat. Hier muss ein Verzeichnis mit dem Namen „SHOUTcast“ existieren, indem alle benötigten Dateien abgelegt werden die für den Aufruf des Flash Players benötigt werden.

```
cd /home/streaming/public_html
```

```
mkdir SHOUTcast
```

```
cp -r /usr/local/WowzaMediaServerPro/examples/SHOUTcast/client SHOUTcast
```

Nach dem Kopieren der notwendigen Client-Dateien, sollte sich der Flash Player über die URL *http://SERVER-URL/streaming/SHOUTcast/client/shoutcast.html* zeigen.

Wenn der Wowza Media Server gestartet wurde und ein Icecast Server einen Stream bereitstellt, dann kann der Flash Player getestet werden. Dazu benötigt man lediglich die URL des Streams, in dieser Testumgebung streamt ein Icecast Server einen Stream mit dem Namen *stream* auf den Mountpoint *http://192.168.1.33:8000/stream*, und die URL auf dem der Wowza Streaming Server läuft, in dieser Testumgebung läuft er auf der IP *192.168.1.33* und den Namen der Anwendung, in dem Beispiel *shoutcast*.

Im Flash Player müssen folgende Angaben gemacht werden:

Server: rtmp://192.168.1.33/shoutcast
URL: http://192.168.1.33:8000/stream.

7.4 Einfügen von Metadaten in die Audio Streams

Um das Einfügen und das Aktualisieren der Metadaten innerhalb eines Streams beispielhaft zu erläutern, wird in den folgenden Abschnitten ein Workaround vorgestellt, welcher den Namen des Radiosenders, die aktuelle Uhrzeit und das aktuelle Datum in den Stream schreibt und diese Information jede Sekunde aktualisiert.

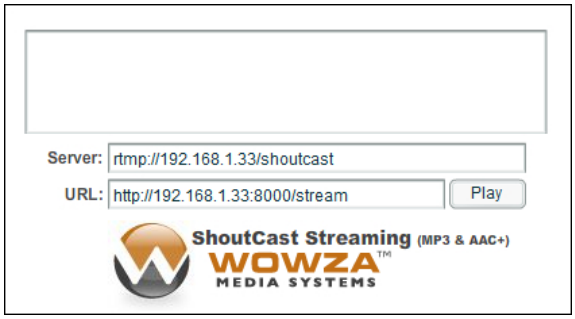


Abbildung 24: Der mitgelieferte Shoutcast Flash Player von Wowza

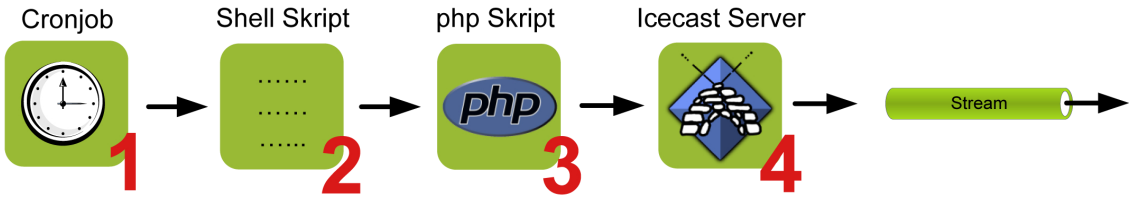


Abbildung 25: Workaround für die Integration einer laufenden Uhrzeit

Um einen solchen Workaround zu erstellen, sind diverse Shell- und PHP-Skripte von Nöten die den Ablauf steuern und die Aktualisierung durchführen. Das folgende Beispiel verwendet ein Shellskript und pro Stream jeweils eine PHP-Datei. Diese Dateien müssen in die bereits existierende Verzeichnisstruktur des Users „streaming“ mit eingepflegt werden.

▸ /home/streaming/meta4stream/bin/

In diesem Verzeichnis liegen die benötigten Shellskripte für die Aktualisierung. In diesem Beispiel ist es nur das Shellskript „time2icecast.sh“.

▸ /home/streaming/public_html/time2icecast/

In diesem Verzeichnis, worauf ein Webserver Zugriff hat, liegen die benötigten PHP-Skripte.

7.4.1 Der Cronjob

Der Cronjob hat innerhalb dieses Workaround mehrere Aufgaben. Er dient dazu um bei einem Start des Servers den Workaround zu starten und ihn ebenfalls jede Minute neu zu triggern. Dabei ruft er kontinuierlich das Shellskript „time2icecast.sh“ auf. Durch diesen Prozess wird gewährleistet, dass der Workaround nie zum erliegen kommt.

Bei einem evtl. auftretenden Fehlerfall, werden die Streams nach maximal einer Minute wieder mit Metadaten versorgt. Bit dem Befehl „crontab -e“ können alle Cronjobs, die auf dem System vorhanden sind, angezeigt und bearbeitet werden.

Der Cronjob:

```
*/1 * * * * sh /home/streaming/metadata4icecast/bin/time2icecast.sh
```

Mit dem oben stehendem Eintrag, wird das Shellskript „time2icecast.sh“ jede Minute erneut gerufen.

7.4.2 Das Shellskript

Das Shellskript ist die zentrale Steuereinheit in diesem Prozeß. Dieses Skript ruft das PHP-Skript, welches die Aktualisierung durchführt, mit den notwendigen Parametern auf. In diesem Beispiel wird nur ein Stream mit neuen Metadaten versorgt. Das Shellskript ist aber in der Lage mehrere Streams zu versorgen.

▸ /home/streaming/meta4stream/bin/time2icecast.sh

Im folgendem Abschnitt wird der Quelltext des Shellskript aufgeführt und wichtige Parameter erläutert.

time2icecast.sh:

```
#!/bin/bash
SCRIPT_URL='http://192.168.1.33/~streaming/time2icecast/'
ADMINPWD='letsstream'
ICECAST_IP="192.168.1.33"
ICECAST_PORT='8000'
MOUNTPOINTS=("stream")
MOUNTPOINTSDESC=('Radio Mittweida')
RUNSPERMINUTE="60"
MAXMINUTEAVERAGE="2"

i=0
for MOUNTPOINT in ${MOUNTPOINTS[@]}
do
$SCRIPT_URL"metadata_"$MOUNTPOINT".php?icecast_ip=$ICECAST_IP&icecast_port=$ICECAST_PORT&mountpoint=$MOUNTPOINT& \
mountpointdesc=${MOUNTPOINTSDESC[$i]}&adminpwd=$ADMINPWD&runsperminute=$RUNSPERMINUTE& \
maxminuteaverage=$MAXMINUTEAVERAGE"

/usr/bin/wget  --timeout=1 $SCRIPT_URL"metadata_"$MOUNTPOINT".php?icecast_ip=$ICECAST_IP&icecast_port=$ICECAST_PORT& \
mountpoint=$MOUNTPOINT&mountpointdesc=${MOUNTPOINTSDESC[$i]}&adminpwd=$ADMINPWD& \
runsperminute=$RUNSPERMINUTE&maxminuteaverage=$MAXMINUTEAVERAGE"
# -O /dev/null > /dev/null &
i=$((i+1))
done
```

SCRIPT_URL: Die URL unter der die PHP-Dateien auf dem Webserver errichbar sind.

ADMINPWD: Das Admin Password des Icecast Server.

ICECAST_IP: Die IP-Adresse unter der der Icecast Server zu erreichen ist.

ICECAST_PORT: Der Port unter dem der Icecast Server seine Streams verbreitet.

MOUNTPOINTS: Ein Array indem alle Namen der Streams, die versorgt werden sollen, stehen.

MOUNTPOINTSDESC: Ein Array indem die Namen, die in die Metadaten des jeweiligen Streams eingefügt werden sollen, stehen.

RUNSPERMINUTE: Gibt an wie oft der Updateprozeß in einer Minute durchgeführt werden soll.

7.4.3 Das PHP-Skript

Das PHP-Skript schreibt die von dem Shellscript erhaltenen Daten, wie zum Beispiel den Name des Radiosenders, und die aktuelle Uhrzeit und das Datum in den Stream. Für jeden Stream der in der „time2icecast.sh“ angegeben ist, muss eine solche PHP-Datei existieren. In diesem Beispiel wird nur der Stream mit dem Namen „stream“ mit Metadaten versorgt, daher ist auch nur eine PHP-Datei mit dem Namen „meta_stream.php“ von Nöten.

/home/streaming/public_html/time2icecast/meta_stream.php

Der Quellcode dieser PHP-Datei ist im Anhang zu finden.

Das Resultat

Abbildung 26: Darstellung der Metadaten in „Winamp“ und „Flash Player“

Im Anschluss stehen innerhalb der Player die Metadaten, in diesem Workaround der Name des Radiosenders und die aktuelle Uhrzeit/Datum, zur Verfügung.

8 Zusammenfassung

Ziel dieser Diplomarbeit war es, mögliche Lösungsansätze für die Präsentation von Audio- und Videodaten auf einer Portalseite aufzuzeigen. Nach der Vorstellung wichtiger Grundlagen und Begriffe, die das Thema Streaming betreffen, wurden verschiedene Lösungsansätze und Techniken erarbeitet.

Anhand der Auswertung einer Onlineumfrage mit 60 Befragten Teilnehmern und der aufgestellten Anforderungen an ein solches Streamingsystem, wurden diverse Lösungen genauer betrachtet und stellten somit die Basis für die Erarbeitung der prototypischen Umsetzung dar.

Durch die so entstandenen Konzepte und Workarounds, ist es möglich einen Audio-Live-Stream von einer Echtzeit-Audioquelle zu verschiedenen Media Playern und Flash Playern über das Internet zu übertragen. Zusätzlich wurde eine mögliche Lösung für die Inbetriebnahme einer automatischen Aktualisierung der Metadaten innerhalb eines Streams entworfen und beispielhaft erklärt.

Durch das diskutieren und analysieren von verschiedenen Lösungswegen zur Realisierung eines Video-On-Demand und Video-Live-Streaming Systems, ist ein möglicher Ansatz für die Verbreitung von Video-Clips und eines TV-Signals über eine Portalseite dargestellt wurden.

Das Ziel der Diplomarbeit, dass finden eines Lösungsansatzes für das Verbreiten von Audio- und Videoinhalten innerhalb eines Redaktionssystems, ist somit erreicht worden.

A Anhang

A.1 Mögliche DABiS XML-Outputs

A.1.1 Standard XML-Output einer DABiS800 Plattform

XML-Output DABiS800:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DATAPACKET Version="2.0">
  <METADATA>
    <FIELDS>
      <FIELD attrname="SEID" fieldtype="i4"/>
      <FIELD attrname="SESNID" fieldtype="i4"/>
      <FIELD attrname="SETIID" fieldtype="i4"/>
      <FIELD attrname="SEDATUM" fieldtype="dateTime"/>
      <FIELD attrname="SESPPOSITION" fieldtype="i4"/>
      <FIELD attrname="SESNPOSITION" fieldtype="i4"/>
      <FIELD attrname="SEPOSITION" fieldtype="i4"/>
      <FIELD attrname="SEZEIT" fieldtype="i4"/>
      <FIELD attrname="SEDAUER" fieldtype="i4"/>
      <FIELD attrname="SESEQUENZ" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SEGRUPPE" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SEKLAMMER" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SEPARALLEL" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SERFLAG" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SETYP" fieldtype="string" WIDTH="2"/>
      <FIELD attrname="SETITELTHEMA" fieldtype="string" WIDTH="50"/>
      <FIELD attrname="SEINTERPRET" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SEKURZBEZ" fieldtype="string" WIDTH="4"/>
      <FIELD attrname="SEGESCHLECHT" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SEJAHRGANG" fieldtype="i2"/>
      <FIELD attrname="SEAUTOR" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SEARCHIVNR" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SESYSTEMNR" fieldtype="string" WIDTH="10"/>
      <FIELD attrname="SEERSTELLDATUM" fieldtype="dateTime"/>
      <FIELD attrname="SEERSTELLER" fieldtype="string" WIDTH="15"/>
      <FIELD attrname="SEBEARBDATUM" fieldtype="dateTime"/>
      <FIELD attrname="SEBEARBEITER" fieldtype="string" WIDTH="15"/>
      <FIELD attrname="SEQUELLE" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SEANMODERATION" fieldtype="bin.hex" SUBTYPE="Text" WIDTH="1"/>
      <FIELD attrname="SETEXT" fieldtype="bin.hex" SUBTYPE="Text" WIDTH="1"/>
      <FIELD attrname="SEABMODERATION" fieldtype="bin.hex" SUBTYPE="Text" WIDTH="1"/>
      <FIELD attrname="SEZUSATZTEXT" fieldtype="bin.hex" SUBTYPE="Text" WIDTH="1"/>
      <FIELD attrname="SEZUSATZFELD1" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SEZUSATZFELD2" fieldtype="string" WIDTH="80"/>
      <FIELD attrname="SESCHEMANR" fieldtype="string" WIDTH="15"/>
      <FIELD attrname="SEANMODSTATUS" fieldtype="i2"/>
      <FIELD attrname="SEABMODSTATUS" fieldtype="i2"/>
      <FIELD attrname="SETEXTSTATUS" fieldtype="i2"/>
      <FIELD attrname="SEDROPINLEVEL" fieldtype="i4"/>
      <FIELD attrname="SEFADEINCHAR" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SEFADEOUTCHAR" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SEOUTROCHAR" fieldtype="string" WIDTH="1"/>
      <FIELD attrname="SEISTDATUM" fieldtype="dateTime"/>
      <FIELD attrname="SEISTZEIT" fieldtype="i4"/>
      <FIELD attrname="SEISTDAUER" fieldtype="i4"/>
      <FIELD attrname="SESTATUS" fieldtype="i2"/>
      <FIELD attrname="SEFILENAME" fieldtype="string" WIDTH="255"/>
      <FIELD attrname="SEBILDNAME" fieldtype="string" WIDTH="255"/>
      <FIELD attrname="SEAUDIOFLAG" fieldtype="i2"/>
      <FIELD attrname="SNPOSITION" fieldtype="i4"/>
      <FIELD attrname="SNZEIT" fieldtype="i4"/>
      <FIELD attrname="SNBEZEICHNUNG" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SNMODERATOR1" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SNMODERATOR2" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SNMODERATOR3" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SNMODERATOR4" fieldtype="string" WIDTH="30"/>
      <FIELD attrname="SNSTATUS" fieldtype="i2"/>
    </FIELDS>
    <PARAMS/>
  </METADATA>
  <ROWDATA>
    <ROW SEID="221184" SEDATUM="20080414" SEZEIT="48805104" SEDAUER="307220"
    SEGRUPPE="-" SETYP="M" SETITELTHEMA="Maggie May" SEINTERPRET="Stewart, Rod"
```

```
    SEAUTOR="" SEARCHIVNR="M3990" SEISTDATUM="20080414" SEISTZEIT="48805104"
    SEISTDAUER="307220" SEFILENAME="D:&#092;DABIS800&#092;D800_DAT&#092;MUSIK&#092;6776.MUS"
    SNBEZEICHNUNG="13 Radio 1 am Mittag" SNMODERATOR1="Sabine Renz"/>
    <ROW SEID="224316" SEDATUM="20080414" SEZEIT="48575475" SEDAUER="210744" SEGRUPPE="-"
    SETYP="M" SETITELTHEMA="SHE DRIVES ME CRAZY" SEINTERPRET="FINE YOUNG CANNIBALS"
    SEAUTOR="Steele/Gift" SEARCHIVNR="M2174" SEISTDATUM="20080414" SEISTZEIT="48575475"
    SEISTDAUER="210744" SEFILENAME="D:&#092;DABIS800&#092;D800_DAT&#092;MUSIK&#092;5862.MUS"
    SNBEZEICHNUNG="13 Radio 1 am Mittag" SNMODERATOR1="Sabine Renz"/>
    <ROW SEID="221182" SEDATUM="20080414" SEZEIT="48344286" SEDAUER="231192"
    SEGRUPPE="-" SETYP="M" SETITELTHEMA="SIE IST WEG" SEINTERPRET="FANTASTISCHEN VIER"
    SEAUTOR="" SEARCHIVNR="M1176" SEISTDATUM="20080414" SEISTZEIT="48344286"
    SEISTDAUER="231192" SEFILENAME="D:&#092;DABIS800&#092;D800_DAT&#092;MUSIK&#092;4946.MUS"
    SNBEZEICHNUNG="13 Radio 1 am Mittag" SNMODERATOR1="Sabine Renz"/>
    <ROW SEID="212870" SEDATUM="20080414" SEZEIT="46948876" SEDAUER="6330"
    SEGRUPPE="G" SETYP="J" SETITELTHEMA="Wetter" SEINTERPRET="" SEAUTOR=""
    SEARCHIVNR="R1 Wetter" SEISTDATUM="20080414" SEISTZEIT="46948876" SEISTDAUER="6330"
    SEFILENAME="D:&#092;DABIS800&#092;D800_DAT&#092;VERPACKUNGEN&#092;WETTER&#092;31726.MUS"
    SNBEZEICHNUNG="13 Radio 1 am Mittag" SNMODERATOR1="Sabine Renz"/>
  </ROWDATA>
</DATAPACKET>
```

A.1.2 XML-Output mit Platzhaltern einer DABiS800 Plattform

XML-Output DABiS800:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DATAPACKET Version="2.0">
  <ROWDATA>
    <ROW>
      <NEXT>1</NEXT>
      <ID>14152</ID>
      <ARCHIVNR>Teaser 16 Uhr</ARCHIVNR>
      <SYSTEMNR></SYSTEMNR>
      <DATUM>2008-10-12T16:00:00+00:00</DATUM>
      <ZEIT>16:00:00+00:00</ZEIT>
      <DAUER>0</DAUER>
      <GRUPPE>A</GRUPPE>
      <TYP>Z</TYP>
      <TITELTHEMA>Und in dieser Stunde...</TITELTHEMA>
      <INTERPRET>16 Uhr</INTERPRET>
      <AUTOR></AUTOR>
      <FILENAME>14152.mus</FILENAME>
      <BEZEICHNUNG>Wochenende</BEZEICHNUNG>
      <MODERATOREN>
        <MODERATOR>Hilgers</MODERATOR>
      </MODERATOREN>
      <MULTIMEDIA>
        <AUDIOS></AUDIOS>
        <VIDEOS></VIDEOS>
        <IMAGES></IMAGES>
      </MULTIMEDIA>
    </ROW>
    <ROW>
      <ID>2346</ID>
      <ARCHIVNR>0004c5a4</ARCHIVNR>
      <SYSTEMNR></SYSTEMNR>
      <DATUM>2008-10-12T15:56:02+00:00</DATUM>
      <ZEIT>15:56:02+00:00</ZEIT>
      <DAUER>173</DAUER>
      <GRUPPE>A</GRUPPE>
      <TYP>M</TYP>
      <TITELTHEMA>SWEETEST THING</TITELTHEMA>
      <INTERPRET>U2</INTERPRET>
      <AUTOR>U2</AUTOR>
      <TEXT>
        <![CDATA[!!U2]]>
      </TEXT>
      <FILENAME>2346.MUS</FILENAME>
      <BEZEICHNUNG>Wochenende</BEZEICHNUNG>
      <MODERATOREN>
        <MODERATOR>Hilgers</MODERATOR>
      </MODERATOREN>
      <MULTIMEDIA>
        <AUDIOS></AUDIOS>
        <VIDEOS></VIDEOS>
```

```
        <IMAGES></IMAGES>
      </MULTIMEDIA>
    </ROW>
    <ROW>
      <ID>51630</ID>
      <ARCHIVNR></ARCHIVNR>
      <SYSTEMNR></SYSTEMNR>
      <DATUM>2008-10-12T15:30:02+00:00</DATUM>
      <ZEIT>15:30:02+00:00</ZEIT>
      <DAUER>1330</DAUER>
      <GRUPPE>A</GRUPPE>
      <TYP>B</TYP>
      <TITELTHEMA>SpvGG Greuther FÃ¼rth - FC St. Pauli</TITELTHEMA>
      <INTERPRET>2. HZ B</INTERPRET>
      <AUTOR></AUTOR>
      <FILENAME>51630.MUS</FILENAME>
      <BEZEICHNUNG>Wochenende</BEZEICHNUNG>
      <MODERATOREN>
        <MODERATOR>Hilgers</MODERATOR>
      </MODERATOREN>
      <MULTIMEDIA>
        <AUDIOS></AUDIOS>
        <VIDEOS></VIDEOS>
        <IMAGES>
          <FILE>
            <FILENAME>51630.01.jpg</FILENAME>
            <MIMETYPE>image/jpeg</MIMETYPE>
            <FTP>
              <!-- FTP SERVER -->
            </FTP>
            <RELATIVEPATH>51630.01.jpg</RELATIVEPATH>
            <ABSOLUTEPATH></ABSOLUTEPATH>
            <TIMER>0</TIMER>
            <BAUCHBINDE></BAUCHBINDE>
            <BAUCHBINDE_FUNKTION></BAUCHBINDE_FUNKTION>
          </FILE>
        </IMAGES>
      </MULTIMEDIA>
    </ROW>
  </ROWDATA>
</DATAPACKET>
```

A.1.3 Möglicher XML-Output per NXC

NXC:

```
<?xml version="1.0" encoding="UTF-8"?>
<station name="Aby Oldie">
 <onair>
  <title>1 2 3 4</title>
  <artist>COOLIO</artist>
  <type>M</type>
  <duration>03:05</duration>
  <start>2010-01-15 10:53:42</start>
  <date>2010-01-15 10:53:42</date>
  <cover>271978.jpg</cover>
  <moderator>AVO</moderator>
 </onair>
 <next>
  <title> </title>
  <artist> </artist>
  <type> </type>
  <duration>00:00</duration>
  <start>1899-12-30 00:00:00</start>
 </next>
<mytag>1 2 3 4 -- COOLIO </mytag>
</station>
```

A.2 Aktualisierung der Metadaten

metadata_stream.php:

```
<?php
$access_file= "icemeta_".$_REQUEST['mountpoint']."_in_access.bin";
$max_age_off_access_file=120;
$access_file_checks= 20;

function http_get( $host, $port, $request="/ HTTP/1.0" ) {

$fp = fsockopen($host, $port);
  if (!$fp) {
return false;
  } else {
    fwrite($fp, "GET $request\r\n\r\n");
    stream_set_timeout($fp, 3);
    $res["data"] = fread($fp, 2000);

    $res["meta"] = stream_get_meta_data($fp);
    fclose($fp);

    if ($res["meta"]['timed_out']) {
            return false;
    } else {
            return $res;
    }
  }

}

function setmeta(){

$host = $_REQUEST["icecast_ip"];
$port = $_REQUEST["icecast_port"];
$admin_password=$_REQUEST["adminpwd"];

$song = $_REQUEST["song"];

$defaultmount="stream";
$mountpoint = $_REQUEST["mountpoint"];

if ( $mountpoint != "ALL" ) {
  if ( $mountpoint == "" ) {
$mountpoint = "$defaultmount";
  }
  $mountlist = array( $mountpoint );
}

$song = date("H:i:s d.m.Y");
$song = "".$song." ";
$song = $_REQUEST['mountpointdesc']." | ".$song." ";
$song = utf8_encode($song);
$song = str_replace(" ","%20",$song);

foreach( $mountlist as $mountpoint ) {
  $request="/admin/metadata?mode=updinfo&mount=/$mountpoint&song=$song HTTP/1.0\r\n";
  $request.="User-Agent: RDSupdater (Mozilla compatible)\r\n";
  $request.="Authorization: Basic ";
  $request.= base64_encode("admin:$admin_password");
  $request.="\r\n";

  $statsraw = http_get( $host, $port, $request );
}
}

function doProcess() {
    setmeta();
    return false;
}

function manageProcess() {
    $runsPerMinute = $_REQUEST['runsperminute'];
    $maxMinuteAverage = $_REQUEST['maxminuteaverage'];
    $waitIfNotWorking = 0; // seconds

    $microsPerSecond = 1000000;
```

```
    $currentMinute = 0;
    $minute = date("i");
    $countPerMinute = array();
    $sumPerMinute = array();

    $totalProcessTime = 0;
    $totalCounts = 0;

    while (true) {
        $timestart = microtime();
        $performedWork = doProcess();
        $timeend = microtime();

            $ts = split(" ",$timestart);
            $te = split(" ",$timeend);

            $te[0] = ($te[0] * $microsPerSecond) - ($ts[0] * $microsPerSecond);
            $te[1] = ($te[1] - $ts[1]) * $microsPerSecond;

            $processTime = $te[0] + $te[1];

echo "<br>".$processTime."<br>";

            if (date("i")<>$minute) { // We are NOT in the same minute
                $currentMinute = ($currentMinute+1) % $maxMinuteAverage;
                if (isset($countPerMinute[$currentMinute])) {
                    $totalProcessTime = $totalProcessTime - $sumPerMinute[$currentMinute];
                    $totalCounts = $totalCounts - $countPerMinute[$currentMinute];
                }

                $countPerMinute[$currentMinute] = 0;
                $sumPerMinute[$currentMinute] = 0;

            }

            $countPerMinute[$currentMinute] = $countPerMinute[$currentMinute] + 1;
            $sumPerMinute[$currentMinute] = $sumPerMinute[$currentMinute] + $processTime;

            $totalCounts = $totalCounts + 1;
echo "<br>".$totalCounts."<br><br>";
            $totalProcessTime = $totalProcessTime + $processTime;
            $averageRuntime = round($totalProcessTime / $totalCounts);
            $waitTime = (($microsPerSecond*60) / $runsPerMinute) - $averageRuntime;

if (date("i")<>$minute) {
$performedWork = doProcess();
break;
}
else{
if ($waitTime > 0){
usleep($waitTime);
}
else {
$performedWork = doProcess();
break;
}
}
    }

}

function check_access_file(){

global $max_age_off_access_file;
global $access_file;

if ( file_exists ( $access_file ) ) {

$if_access_file = true;
$access_file_date = filectime($access_file);
$meindatumsformat = date(' m.d.y', $access_file_date);
$meindatumsformat = date(microtime(), $access_file_date);

$age_off_access_file = time()- $access_file_date;

if ($age_off_access_file >= $max_age_off_access_file){
unlink ( $access_file );
return false;
```

```
}
else {
return true;
}
}
else {
return false;
}
}

$access_file_exist = check_access_file();

if ($access_file_exist == true){

for($count = 0; $count < $access_file_checks; $count++){
usleep(250000);
clearstatcache();
$access_file_exist = check_access_file();
if ($access_file_exist != true){
$f = fopen( $access_file , 'w' );
fclose ( $f );
$hijacked = true;
break;
}
}
if ($access_file_exist == true){
return;
}

}
if (($access_file_exist != true) || $hijacked){
if (!$hijacked){
$f = fopen( $access_file , 'w' );
fclose ( $f );
}
manageProcess();
unlink ( $access_file );
}

?>
```

A.3 Abbildungen der einzelnen Seiten der Umfrage

A.3.1 Sprachauswahl und Startseite

Abbildung 27: Sprachauswahl (oben) und Startseite (unten)

A.3.2 Gewinnspiel und erste Frage

Abbildung 28: Gewinnspielseite (oben) und die ersten Fragen (unten)

B Abkürzungverzeichnis

AAC	Advanced Audio Coding
ARD	Arbeitsgemeinschaft der öffentlich-rechtlichen Rundfunkanstalten der Bundesrepublik Deutschland
AS2	Actionscript Version 2
AS3	Actionscript Version 3
ASP	Active Server Pages
CD	Compact Disc
DABiS	Digital Audio Broadcasting Integrated Solutions
DB	Data Base
DNS	Domain Name System
FDDI	Fiber Distributed Data Interface
FLV	Flash Video
FTP	File Transport Protocol
HTTP	Hypertext Transport Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol
IGMP	Internet Group Management Protocol
IP	Internet Protocol
IPX	Internetwork Packet Exchange
Kbps	Kilobit pro Sekunde
khz	Kilo Hertz
LCD	Liquid Crystal Display
LDAP	Lightweight Directory Access Protocol
MMS	Microsoft Media Server Protocol
MMST	Microsoft Media Server Protocol Tunneled
MP3	MPEG-1 Layer 3
MPEG	Moving Picture Experts Group
MSN	The Microsoft Network
NCP	Network Control Protocol
NXC	Not eXactly C
OS	Operation System
OSI-7	Open Systems Interconnection Reference Model
PC	Personal Computer
PHP	Hypertext Preprocessor
RJ-45	Registered Jack (genormte Buchse)
RTCP	Real-Time Control Protocol
RTMPT	Real Time Messaging Protocol Tunneled
RTP	Real-Time Transport Protocol

RTSP	Real-Time Streaming Protocol
SCTP	Stream Control Transmission Protocol
SMTP	Simple Mail Transfer Protocol
SPX	Sequenced Packet Exchange
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TTL	Time-To-Live
TV	Television
UDP	User Datagram Protocol
UHF	Ultra-High-Frequency
URL	Uniform Resource Locator
WMA	Windows Media Audio
WMV	Windows Media Video
WWW	World Wide Web
XML	Extensible Markup Language
ZDF	Zweites Deutsches Fernsehen

C Glossar

Broadcast: Broadcast-Verbindungen bezeichnen einen Übertragung von einem Punkt an alle Teilnehmer einer Gruppe.

Client: Ein Client ist ein Computer oder ein Computerprogramm, welcher Kontakt zu einem anderen Computer oder Computerprogramm, dem Server, aufnimmt um dessen Dienstleistung zu nutzen.

Cronjob: Eine Jobsteuerung von Unix und unixartigen Betriebssystemen, die wiederkehrende Aufgaben automatisch zu einer bestimmten Zeit ausführen kann.

DABiS: Eine Sendeautomations-Plattform für die Verarbeitung von Content in Radio- und Fernsehstationen.

Download: Es wird damit die Übertragung von Daten von einem Computer in einem Netzwerk oder im Internet zum eigenen Computer (Client) bezeichnet.

Encoder: Ein Encoder ist ein System, das eine Datenquelle (zum Beispiel ein digitales Audiosignal) in ein für einen bestimmten Kanal geeignetes Format umwandeln soll.

Live-Streaming: Eine Form des Streamings, bei dem die zu übertragenden Daten live (bzw. mit einer kleinen Zeitverzögerung durch das Aufbereiten und Senden der Daten) gesendet und empfangen werden können.

Metadaten: Als Metadaten werden beim Streaming Daten bezeichnet, die den Inhalt eines Audio-oder Video-Streams beschreiben.

Multicast: Multicast-Verbindungen bezeichnen in der Telekommunikation eine Nachrichtenübertragung von einem Punkt zu einer Gruppe.

On-Demand-Streaming: Eine Form des Streamings, bei dem die Daten auf dem Server bereitliegen und dann gesendet werden wenn der Benutzer sie anfragt.

Progressiver Download: Die Übertragung von Daten im Internet ähnlich dem Download, nur das hierbei die Daten bereits während dem Herunterladen genutzt werden können.

Server: Als Server wird entweder eine Software im Rahmen des Client-Server-Modells oder eine Hardware, auf der diese Software läuft bezeichnet.

Sourceclient: Als Sourceclient wird eine Soft- oder Hardware bezeichnet, die neben dem Kodieren eines Signales auch andere Arbeiten wie das Komprimieren und das Bezeichnen des Signales mit ausführen.

Streaming: Den Vorgang der Datenübertragung, bei dem gleichzeitigen empfangen und wiedergeben von Audio- und Videodaten aus einem Rechnernetz, nennt man Streaming.

Unicast: Unicast-Verbindungen sind Verbindungen, an denen genau zwei Kommunikationspartner beteiligt sind.

D Literaturverzeichnis

(ADOBE ONLINE 1): Verfügbar bei: http://www.adobe.com/devnet/rtmp/

(BR-ONLINE): Verfügbar bei: http://www.br-online.de/br-intern/medienforschung/onlinenutzung/onlinestudie (20.01.2010)

(BLACK U.): Uyless Black: Internettechnologien der Zukunft, Addison-Wesley 1999

(WEGNER R.): Wegner Ralf, Claus Bachmeier: Streaming Media im Business-Bereich, Addison-Wesley Verlag 2000

(MEIßNER K.): Meißner, Klaus: Internet - Rundfunk, Vistas Verlag Berlin

(LÖTZSCH S.): Lötzsch Steffen: Datenübertragung per RTSP, Februar 2002, Chemnitz, Technische Universität, Fakultät für Informatik, Diplomarbeit

(FISCHER H.): Heike Fischer: Spezifikation und Implementierung eines Streaming-Servers für multimediale Archivinhalte, 1998, Köln, Fachhochschule Köln, Diplomarbeit

(MICROSOFT ONLINE 1): Verfügbar bei: http://msdn.microsoft.com/en-us/library/cc234711%28PROT.10%29.aspx

(MICROSOFT ONLINE 2): Verfügbar bei: http://www.microsoft.com/windows/windowsmedia/howto/articles/introhosting.aspx#quickstartondemand

(MIRCOSOFT ONLINE 3): Verfügbar bei: http://www.microsoft.com

(REAL ONLINE 1): Verfügbar bei: http://www.realnetworks.com/products-services/helix/media/server_proxy.aspx

(WIKI ONLINE 1): Verfügbar bei: http://de.wikipedia.org/wiki/Winamp

(TERRATEC ONLINE 1): Verfügbar bei: http://www.terratec.net

(WOWZA ONLINE 1): Verfügbar bei: http://www.wowzamedia.com/

(WOWZA ONLINE 2): Verfügbar bei: http://www.wowzamedia.com/quickstart.html#intro-supportedmedia

(ICECAST ONLINE 1): Verfügbar bei: http://www.icecast.org/docs.php

E Eigenständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig und unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

....

Bearbeitungsort, Datum Unterschrift